IBM RT PC Advanced Interactive Operating System Version 2.1

AIX Operating System Communications Guide

Communications Family

Personal
Computer
Software

SC23-0791-0

IBM RT PC Advanced Interactive Operating System Version 2.1

AIX Operating System Communications Guide

Communications Family

Personal
Computer
Software

First Edition (January 1987)

This edition applies to Version 2.1 of the AIX Operating System and to all subsequent releases until otherwise indicated in new editions or technical newsletters. Changes are made periodically to the information herein; these changes will be reported in technical newsletters or in new editions of this publication.

References in this publication to IBM products, programs, or services do not imply that IBM intends to make these available in all countries in which IBM operates. Any reference to an IBM licensed program in this publication is not intended to state or imply that only IBM's licensed program may be used. Any functionally equivalent program may be used instead.

International Business Machines Corporation provides this manual "as is," without warranty of any kind, either express or implied, including, but not limited to, the implied warranties of merchantability and fitness for a particular purpose. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this manual at any time.

Products are not stocked at the address given below. Requests for copies of this product and for technical information about the system should be made to your authorized IBM RT PC dealer or your IBM marketing representative.

A reader's comment form is provided at the back of this publication. If the form has been removed, address comments to IBM Corporation, Department 997, 11400 Burnet Road, Austin, Texas 78758. IBM may use or distribute whatever information you supply in any way it believes appropriate without incurring any obligation to you.

About This Book

About This Book

This guide explains how to use the AIX-based[1] communication facilities of the IBM RT Personal Computer[1] .

Who Should Read This Book

This guide is intended for system users who are familiar with the IBM RT PC[1] and the AIX operating system.

In some cases, instruction is divided into basic operations and more advanced topics. System users who need to use a communication facility need only the basic information. Individuals with more advanced knowledge may need to know about customization and configuration.

See "How To Use This Guide" on page vi to decide which chapters to read.

Before You Begin

Before you can use any communication facility described in this book, you must have the appropriate hardware and software. These requirements are described in three books: *IBM RT PC Planning Guide*, *Options Installation*, and *Installing and Customizing the AIX Operating System.*

---

1 AIX, RT Personal Computer, RT PC and RT are trademarks of International Business Machines Corporation.

How To Use This Guide

The preface and Chapter 1 provide an overview that everyone should read. Chapters 2, 3, 4, 5, and 7 each provide basic instructions for one communication facility. General users should read the chapters that discuss the communication facility they want to use.

Chapters 6 and 8 each contain information on the customization and configuration of one communication facility. System administrators and others who need this information should read the chapter on each communication facility for which they have responsibility. Chapter 9 applies to any facility that needs a configured port, and should be read as a continuation of both Chapter 6 and Chapter 8. Chapter 9 also provides an overview of ports, modems, and cables.

Consult Appendix A if you need to install any of the communication facilities, and Appendix B if a problem occurs during the installation or use of a communication facility.

- "Chapter 1. Introduction" provides an overview of AIX-based communication facilities for communicating in a local multi-user environment and with other systems.
- "Chapter 2. Creating a Local System Communication Facility" describes ways to communicate with other users on your local system, using commands from the Base System Program and Multi-User Services.
- "Chapter 3. Creating a Remote Communication Facility with Base System Program Commands" discusses sending mail to a remote computer and using the **connect**[2] command that comes with the base program. Discussion of the ATE **connect** command is in Chapter 5.
- "Chapter 4. Using the Interface Program for Use with TCP/IP" describes basic tasks that you can do on a network with the

---

2 Connect is a trademark of INTERACTIVE Systems Corporation.

interface program to Transmission Control Protocol/Internet Protocol.

- "Chapter 5. Using Asynchronous Terminal Emulation (ATE)" describes the basic ATE tasks and provides an overview of the menus and commands you need to communicate with a remote computer system.

- "Chapter 6. Customizing Asynchronous Terminal Emulation (ATE)" describes customization, configuration and other advanced tasks that tailor this program to your specific needs.

- "Chapter 7. Using the UNIX-to-UNIX[3] Copy Program (UUCP)" describes how to use this background process to send and receive files, run remote commands, and perform several management tasks.

- "Chapter 8. Customizing UUCP" discusses tailoring the UUCP facility to your specific needs.

- "Chapter 9. Using Ports, Cables, and Modems" provides basic information on the configuration of ports and gives a general overview of communication hardware needs.

- "Appendix A. Installing Communications Facilities" gives an overview of the installation processs.

- "Appendix B. Problem Determination" discusses problems that may occur while communication facilities are in use.

- A Reader's Comment Form and Book Evaluation Form are provided at the back of this book. Use the Reader's Comment Form to give IBM information that may improve the book. After you have become familiar with the manual, use the Book Evaluation Form to give IBM specific feedback.

---

3 UNIX was developed and licensed by AT&T. It is a registered trademark of AT&T in the United States of America and other countries.

Conventions

Conventions used in this book are as follows:

- The name of a program or command appears in bold:

  uux program or **uux** command.

- A general name or field for which you must substitute a real name appears in italics:

  filename

- Words that you key in and messages that appear on your display screen are in green:

  ```
  Select an option.
  ```

- Optional flags for commands are enclosed in **[ ]** (brackets). For example, the following line means that you can use the command alone or with the flag:

  ```
  who [-u]
  ```

- Options or commands that may be repeated any number of times are enclosed in **( )** (parentheses).

  ```
  a (commandinitial value)
  ```

- When you are instructed to Enter information, you must type it and then press the **Enter** key.

- When you are instructed to Enter **EOF**, use the key combination indicated on the keyboard reference card for your keyboard. For the RT PC, use **Ctrl-D**.

Prerequisite Information

You can get started in the multi-user environment with relatively little experience in computer communications.

The first facilities discussed in the book (the Local Communication Facility and the Remote Communication Facility with Base System Program Commands) are the least complex.

System-to-system features, such as TCP/IP and Asynchronous Terminal Emulation, require some computer experience and an understanding of how computers communicate with each other. To use UUCP (UNIX-to-UNIX copy facility), you need more knowledge and experience with communications.

You should be familiar with the following publications before you use this guide:

- *IBM RT PC Using the AIX Operating System* describes using the AIX Operating System commands, working with file systems, and developing shell procedures.
- *IBM RT PC Managing the AIX Operating System* provides instructions for performing such system management tasks as adding and deleting user IDs, creating and mounting file systems, and repairing file system damage.
- *IBM RT PC AIX Operating System Commands Reference* lists and describes the AIX Operating System commands.
- *IBM RT PC Planning Guide* provides information for site planning and preparation, software planning, and communications planning. The site planning and preparation information includes some physical specifications of the RT PC system units, devices, and cables. The software planning information includes prerequisites, corequisites, and recommended memory for certain software products. The communications planning information includes prerequisites, corequisites, and network configuration requirements of certain communications products. This book should be consulted before ordering RT PC products.

Related Information

Other publications that you may find helpful include the following:

- *IBM RT PC Messages Reference* lists messages displayed by the IBM RT PC and explains how to respond to the messages.
- *IBM RT PC Guide to Operations* describes the IBM 6151 and IBM 6150 system units, the displays, keyboard, and other devices that can be attached. This guide also includes procedures for operating the hardware and moving the IBM 6151 and IBM 6150 system units.
- *IBM RT PC Installing and Customizing the AIX Operating System* provides step-by-step instructions for installing and customizing the AIX Operating System, including how to add or delete devices from the system and how to define device characteristics. This book also explains how to create, delete, or change AIX and non-AIX minidisks.
- *IBM RT PC Interface Program for use with TCP/IP* describes the Interface Program commands for transferring data among host computers, logging into remote computers, executing commands remotely, and managing networks. This book also describes the programming interfaces to the Interface Program.

Ordering Additional Copies of This Book

To order additional copies of this publication (without program diskettes), use either of the following sources:

- To order from your IBM representative, use Order Number SBOF-0156
- To order from your IBM dealer, use Part Number 79X3847.

A binder, the *AIX Operating System Communications Guide*, and Keyboard Templates are included with the order. For information on ordering the binder, the manual, or the templates separately, contact your IBM representative or your IBM dealer.

Contents

Chapter 1. Introduction 1-1
About This Chapter 1-3
Overview of AIX-Based Communications 1-4
Communication Concepts 1-7

Chapter 2. Creating a Local System Communication Facility 2-1
About This Chapter 2-3
Introduction 2-4
Determining Who Can Receive Messages (who) 2-5
Sending Messages (write) 2-6
Receiving or Rejecting Messages (mesg) 2-9
Conducting an On-Line Conference (confer) 2-11
Sending Mail (mail) 2-18
Receiving Mail (mail) 2-20

Chapter 3. Creating a Remote Communication Facility with Base System Program Commands . . . 3-1
About This Chapter 3-3
Sending Mail to a Remote System (mail) 3-4
Connecting Your Local System to a Remote System (connect) 3-7

Chapter 4. Using the Interface Program for Use with TCP/IP 4-1
About This Chapter 4-3
Before You Begin 4-4
Overview of TCP/IP 4-4
Requesting Information about Users (finger) 4-5
Requesting Information about Remote Systems (ping) 4-7
Transferring Files (xftp) 4-8
Sending Mail (netmail) 4-14
Using a Remote Login (tn) 4-17

Chapter 5. Using Asynchronous Terminal Emulation (ATE) 5-1
About This Chapter 5-3
Overview of Asynchronous Terminal Emulation (ATE) Tasks . 5-4
Giving Commands 5-8
Prerequisite Tasks 5-12
Starting ATE 5-14
Making a Connection (ATE connect) 5-14
Displaying a Dialing Directory (directory) 5-21
Creating a Dialing Directory File 5-23
Sending a File (send) 5-28
Receiving a File (receive) 5-30
Interrupting a Session (break) 5-32
Terminating a Session (terminate) 5-32
Getting Help (help) 5-33
Performing an Operating System Command (perform) 5-35
Leaving ATE (quit) 5-36

Chapter 6. Customizing Asynchronous Terminal Emulation (ATE) 6-1
About This Chapter 6-3
Before You Begin 6-4
Modifying Local Settings (modify) 6-4
Altering Connection Settings (alter) 6-9
Changing (Remapping) the Control Keys 6-15
Changing Your Default File 6-17
Using Xmodem Protocol for File Transfer (xmodem) 6-20
Using Pacing Protocol for File Transfer 6-23

Chapter 7. Using the UNIX-to-UNIX Copy Program (UUCP) 7-1
About This Chapter 7-3
Introduction to UUCP 7-4
Identifying Compatible Systems (uuname) 7-5
Sending and Receiving Files (uucp) 7-7
Controlling File Access at the Local System 7-14
Running Remote Commands (uux) 7-20
Getting Status Information (uulog) 7-23
Removing Old Files (uuclean) 7-24
Establishing a Subnetwork (uusub) 7-26

Running Commands Automatically (uudemon) 7-28
How UUCP Works 7-32
How UUX Works 7-37

Chapter 8. Customizing UUCP 8-1
About This Chapter 8-3
Overview of UUCP Customization and Configuration 8-4
Choosing a Name for Your Site (chparm) 8-5
Editing Files 8-6
Administrative and Problem Determination Files 8-20

Chapter 9. Using Ports, Cables, and Modems 9-1
About This Chapter 9-3
Ports 9-4
Connecting Terminals and Modems 9-7
Modems 9-9

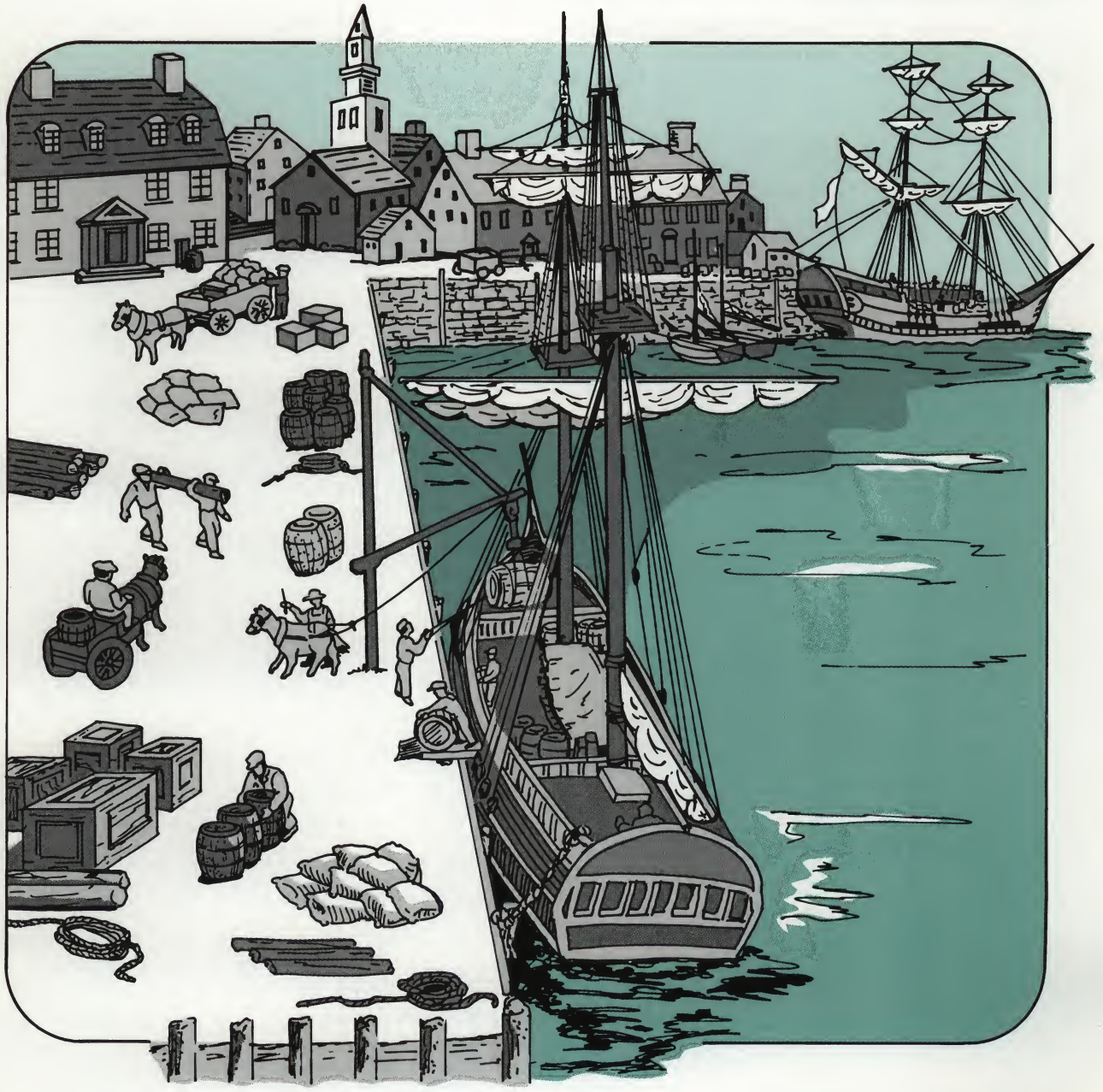
Appendix A. Installing Communications Facilities . A-1
About This Appendix A-3
Installing the AIX Operating System A-4
Installing the Communication Facilities A-5

Appendix B. Problem Determination B-1
Asynchronous Terminal Emulation Problems B-3
UUCP Problems B-10

Glossary X-1

Index X-9

Chapter 1. Introduction



CONTENTS

About This Chapter 1-3
Overview of AIX-Based Communications 1-4
  Local System Communication 1-4
  Remote Communication Using Base System Program Commands 1-5
  Interface Program for Use with TCP/IP 1-5
  Asynchronous Terminal Emulation 1-6
  UNIX-to-UNIX Copy Program (UUCP) 1-6
Communication Concepts 1-7

About This Chapter

This chapter provides an overview of the communication facilities provided with AIX to help you select the best facility for each of your communication tasks.

The facilities discussed here are:

- Local System Communication (using commands from the AIX Base System Program and the Inter-workstation commands component of Multi-User Services).
- Remote System Communication (using commands from the AIX Base System Program).
- Interface Program for Use with TCP/IP
- Asynchronous Terminal Emulation (ATE)
- Unix-to-Unix Copy Program (UUCP).

Overview of AIX-Based Communications

The AIX Operating System contains several communication facilities that help you transfer information electronically from one location to another. Depending on the facility, this transfer can be within a local system or between systems, and involve any of these:

- Two terminals or work stations
- Two computer systems
- A work station and a computer system.

For some facilities, modems and cables are required. These are discussed briefly in "Chapter 9. Using Ports, Cables, and Modems."

To help you decide which facilities you need, refer to the descriptions that follow and to the *IBM RT PC Planning Guide.*

Local System Communication

Several commands in the Base System Program and the Multi-User Services facility provide message-related functions for a local system.

With these commands you can:

- Send and receive messages (**who**, **write**, and **mesg**).
- Send and receive mail (**mail**)
- Conduct on-line conferences (**confer**).

You obtain the Base System Program commands when you install AIX. The Multi-User Services commands are in a component of the facility called Inter-workstation Commands, and must be installed separately. Install just the Inter-workstation Commands by selecting option 3 on the install menu (diskette 1 of Multi-User Services). See Appendix A for a brief discussion of installation procedures.

To use these commands, refer to "Chapter 2. Creating a Local System Communication Facility."

Remote Communication Using Base System Program Commands

The Base System Program for AIX contains two commands for communicating with a remote system.

The **mail** command, in addition to its functions on a local system, can send mail to a remote system.

The **connect** command establishes communication with a remote system, and is discussed in "Connecting Your Local System to a Remote System (connect)" on page 3-7.

Since these commands are an integral part of the Base System Program, you obtain the commands when you install AIX, and no further installation is needed.

These commands are discussed in "Chapter 3. Creating a Remote Communication Facility with Base System Program Commands."

Interface Program for Use with TCP/IP

The Interface Program for Use with TCP/IP is on a separate diskette that comes with the AIX Operating System.

TCP/IP uses the Transmission Control Protocol/Internet Protocol to make a connection to a remote system on a network. You can transfer files, send remote mail, and use a remote login.

The installation procedure for the Interface Program is described briefly in Appendix A.

To use this facility, refer to "Chapter 4. Using the Interface Program for Use with TCP/IP."

Asynchronous Terminal Emulation

The Asynchronous Terminal Emulation (ATE) facility is on an individual diskette that comes with the AIX Operating System.

With ATE you can make a connection to a remote computer system, emulate a remote terminal, and use the commands and programs of the remote system. You can also transfer files.

Your local terminal must be configured as a call-out site.

ATE does not let you send files to a specific user on a remote system, or perform file transfers as a background process so you can do other work at your terminal. For these tasks, the UUCP facility, described in the next section, may better fit your needs.

The installation procedure for Asynchronous Terminal Emulation is described briefly in Appendix A.

To use this facility, refer to "Chapter 5. Using Asynchronous Terminal Emulation (ATE)", "Chapter 6. Customizing ATE," and "Chapter 9. Using Ports, Cables, and Modems."

UNIX-to-UNIX Copy Program (UUCP)

The UNIX-to-UNIX Copy Program facility (UUCP) is a component of the Extended Services feature of the AIX Operating System.

UUCP includes a set of directories, files, programs, and commands to let you communicate with a remote system over a dedicated (hardwired) line or a telephone line. You can configure each system as a call-in site, a call-out site, or both.

UUCP lets you perform file transfer tasks and remote execution in the background so that you can use your work station for other work while these tasks are running.

The tasks you can perform with UUCP include the following:

- Gather and send files from one system to a specific user or destination file on a remote or local system.

- Run commands on another system without logging into it.
- Route file transfers through several intermediate systems.

Since UUCP is complex, you should be familiar with your IBM RT PC and the AIX Operating System before you start.

To install just the **uucp** component of Extended Services, select option 6 on the install menu (diskette 1 of Extended Services).

The installation procedure for UUCP is described briefly in Appendix A.

To use this facility, refer to "Chapter 7. Using the UNIX-to-UNIX Copy Program (UUCP)", "Chapter 8. Customizing UUCP", and "Chapter 9. Using Ports, Cables, and Modems."

Communication Concepts

If you need basic information on electronic communications, refer to IBM publication *Data Communications Concepts* (GC21-5169).

Chapter 2. Creating a Local System Communication Facility

CONTENTS

About This Chapter 2-3
Introduction 2-4
Determining Who Can Receive Messages (who) 2-5
Sending Messages (write) 2-6
  Having a Conversation 2-7
  Sending a Longer Message 2-7
  Retaining the Connection between Work Stations 2-8
  System Errors 2-8
Receiving or Rejecting Messages (mesg) 2-9
  Changing the Start-Up Procedure 2-10
Conducting an On-Line Conference (confer) 2-11
  Arranging a Conference 2-11
  Holding a Conference 2-12
  Withdrawing from a Conference 2-14
  Getting a Transcript 2-16
  Closing the Conference 2-17
Sending Mail (mail) 2-18
  Several Mail Boxes 2-19
Receiving Mail (mail) 2-20
  Forwarding Your Mail 2-20
  Reading Your Mail 2-20
  Handling the Mail 2-22

About This Chapter

This chapter shows you how to create a simple communication facility for your local system so you can:

- Send and receive messages.
- Send and receive mail.
- Conduct on-line conferences.

You perform these tasks using commands from both the AIX Base System Program and the Multi-User Services facility.

The Base System Program is an integral part of AIX, so its commands are installed when you install AIX.

The Multi-User Services commands that you need are in its Inter-workstation Commands component. You must install this component in addition to AIX. Select Inter-workstation Commands (item 3) from the list of choices on the install menu on diskette 1 of Multi-User Services. Then, follow the instructions that appear on the display.

See Appendix A for an overview of installation procedures. If you need additional information, refer to *Installing and Customizing the AIX Operating System.*

Introduction

Using the information in this chapter, you can create a facility for local communication. You do this using the **write** and the **mail** commands in the Base System Program, and the **who**, **mesg**, and **confer** commands in the Inter-workstation commands component of Multi-User Services.

The communications described in this chapter are of three types:

messages The sender and the receiver both are logged into the system.

mail Notes are stored in the receiver's mail box whether or not the receiver is logged in.

conferences Messages are sent back and forth by a pre-arranged protocol.

The rest of this chapter discusses the specific tasks you can perform. You can:

- Determine who can receive messages currently.
- Send messages to a user name logged into the system.
- Receive or reject messages when you are logged into the system.
- Participate in an on-line conference.
- Send mail to a mail box.
- Receive and handle mail in your mail box.

If you want to send mail to a user name on a remote system, refer to the next chapter.

For more detail about the commands mentioned here, see *AIX Operating System Commands Reference.*

Determining Who Can Receive Messages (who)

To receive messages, one must be logged into the local system when the message is sent. You can see who is logged in by using the **who** command.

who Displays the user name and work station of all users who currently are logged in, and shows the date and hour each current session began.

-u Adds the number of hours and minutes since there was activity at a work station, and provides the process ID number of each user. Activity for less than one minute is indicated by a . (dot).

Determining Who Is Logged In

- Type **who** after the $ on the command line.

  ```
  who
  ```

  If system user bjh logged in at 7:43 from work station tty0 and jmc logged in from work station tty1 at 9:27, you see:

  ```
  bjh  tty0  Feb 8  07:43
  jmc  tty1  Feb 8  09:27
  ```

- Type **who -u** for more extensive information. If bjh has been inactive for an hour and two minutes, and jmc has been inactive for less than a minute, you see:

  ```
  bjh  tty00  Feb 8  07:43  01:02  17
  jmc  tty01  Feb 8  09:27    .    28
  ```

  The 17 and 28 are process IDs.

Sending Messages (write)

You can send a message to anyone currently logged into the system by using the **write** command. If the person is not currently logged in, the system sends you this message:

```
user is not logged on
```

If this happens, you can send a note to the receiver's mail box. The receiver will see the note at the next login. See "Sending Mail (mail)" on page 2-18 for information on the **mail** command.

The box that follows shows you how to send a message, using `bjh` as the person to whom the message is sent.

How to Send a Message

1. Type **write** *username* after the `$` prompt on the command line.

   ```
   write bjh
   ```

   This sends an alerting sound (the ASCII BEL character) and a notice to `bjh's` display:

   ```
   message from jmc tty1 Feb 8 10:32:45
   ```

   When the connection is ready, you receive two alerting sounds (ASCII BEL characters).

2. Type your message. Press the **Enter** key.

3. When you finish the message, send an end-of-file (EOF) symbol on a separate line. (On the RT PC console, use **Ctrl-D**. To do this, hold the **Ctrl** key and press the **D** key.)

   This tells `bjh` that you have finished, and signals the system to terminate the connection.

Additional Information

- The EOF symbol that you send at the end of a message depends upon the configuration of your computer keyboard. Refer to the information you received with your system if **Ctrl-D** does not produce an EOF symbol.
- If `bjh` answers your message, you see the following:

```
message from bjh (tty0) [Feb 8 10:33:03]
```

Having a Conversation

If you expect to have a conversation--messages sent back and forth--select a symbol (such as the letter **o**) to tell the receiver when a message is over. Select another symbol to identify the end of the conversation (**oo** for over and out.)

If you use an EOF symbol at the end of a message, the system ends the connection and the next sender must begin again with the **write** command.

Sending a Longer Message

If you want to send a longer message, you can create a file for the text before you forward the message.

Sending a Longer Message

1. Create a file, using the text editor of your choice.
2. Write the text, ending with an EOF symbol.
3. Use the **write** command to send the message. To send a message file named `letter.jmc` to `bjh`, type:

```
write bjh < letter.jmc
```

When the message is sent, the connection between the work stations ends.

Retaining the Connection between Work Stations

To retain the connection between work stations after a message is sent, type an ! (exclamation point) before the **write** command. The ! is the shell escape symbol, and requests that the shell run the **write** command. In this way, a conversation can continue.

Retaining the Connection

1. Set up a file for your text, like the `letter.jmc` in the previous example.
2. Establish a connection by typing the **write** command.
3. After you receive two alerting sounds (ASCII BEL characters), type an ! (exclamation mark), the command name, and the file name.

   ```
   ! write bjh <letter.jmc
   ```

   The shell runs the command and writes the results to `bjh`.

System Errors

If a system error occurs, you see:

```
cannot write to bjh on tty0
```

Receiving or Rejecting Messages (mesg)

You determine whether you can receive messages by using the **mesg** command.

mesg Indicates the message status of your work station. You see either `is y` or `is n`, depending on the current status.

mesg n Prevents incoming messages from appearing on your display unless:

- A person with superuser authority sends the message.
- The sender uses the **mail** command and sends the message to your mail box. See "Sending Mail (mail)" on page 2-18.

mesg y Permits incoming messages to appear on your display.

How To Check and Change Your Message Status

1. Check your message status by typing **mesg** after the $ prompt on the command line. Press the **Enter** key.

   ```
   mesg
   ```

2. To change the setting, type `mesg y` or `mesg n` For example, to reject messages, type:

   ```
   mesg n
   ```

3. Press the **Enter** key.

   A sender will see the message: `Permission denied.`

You can type **mesg y** or **mesg n** without first checking your message status.

Changing the Start-Up Procedure

The shell start-up procedure you received contains a default value that lets you receive messages. You can change the default so messages are not received by adding `mesg n` to the **.profile** in your **HOME** or **login** directory.

How to Change Your Profile

1. Type **cd** to go to your **HOME** or **login** directory.
2. To edit the file, type:

   ```
   e .profile
   ```

   The **.profile** displays.
3. Type

   ```
   mesg n
   ```

   as the bottom line of the **profile**.
4. Press **Alt-D** to exit.

This change in your personal **.profile** overrides the default value.

Conducting an On-Line Conference (confer)

The **confer** command issues a conference call to invited participants. The conference is on the record, with a transcript sent to each participant, unless a ~ (tilde) is typed after the command.

confer (*username***)**
: Issues a call for an on-line conference (with transcripts) to invited participants. For the conference to occur, each participant must accept the conference call by using the **joinconf** command.

confer ~
: Issues a call for an off-line conference (no transcripts) to invited participants. For the conference to occur, each participant must accept the conference call by using the **joinconf** command.

joinconf *caller's username*
: Used by invited participants to accept a conference call.

For more information on **confer**, see *AIX Operating System Commands Reference.*

Arranging a Conference

One person must issue a conference call and the other participants must actively accept the invitation.

Unless you give the conference a specific name, the system assigns your user name. If you called a previous conference, a number is added. For example, the second conference called by jmc is named

```
jmc-1
```

How to Arrange a Conference

1. To issue a conference call, type **confer** (*username*) after the $ prompt on the command line.

   For example, if jmc invites bjh, chr, and doc to a conference, the command looks like this:

   ```
   confer bjh chr doc
   ```

   Each invited participant who is on-line hears two alerting sounds and receives a notice:

   ```
   Please 'joinconf' jmc
   ```

2. To accept a conference call, each invited participant should type: **joinconf** *caller's username.*

   ```
   joinconf jmc
   ```

3. Each user then receives notice that the others have joined the conference, and prepares to participate.

   ```
   bjh has joined
   chr has joined
   doc has joined
   ```

Holding a Conference

To avoid confusion, participants should agree on the procedures in advance:

- One person takes the floor at a time.
- Each person ends comments with an end-signal and yields the floor.

- The other participants wait for an end-signal before requesting the floor.
- Each participant who leaves the conference must follow the established withdrawal procedure.

How to Participate

1. To take the floor, press the **Enter** key once.

   If you are successful, your name appears in [ ] (brackets) so everyone knows you have the floor.

   ```
   [jmc]
   ```

2. When your name appears, type your comments.
3. When you have finished your comments, type an end-signal, such as **o** (over).
4. Press the **Enter** key once to yield the floor.

   Another participant may then request the floor.

Additional Information

A common way to give end-signals follows:

- End each comment with the letter **o** (over).
- End each conversation (group of comments on the same topic) with **oo** (over and out).

Contending for the Floor

If someone else tries to take the floor while you are trying to do so, your name will not appear. If no other person successfully claims the floor, press the **Enter** key again to repeat your request.

If several people claim the floor at the same time, the last name in brackets is successful. The others should immediately yield by pressing the **Enter** key.

Withdrawing from a Conference

If you want to withdraw from a conference, you must be excused by the other participants. In the following example, you are `chr`, a participant who wants to leave the conference.

Withdrawing from a Conference

1. Take the floor by pressing the **Enter** key once.

   If this is successful, your name appears in [ ] (brackets).

   ```
   [chr]
   ```

2. Press the **EOF** control key.

   The other participants see your name in brackets followed by the word BYE.

   ```
   [chr] BYE
   ```

   You see a prompt that lets you accept or change the transcript decision made at the beginning of the conference. To retain the original decision, press the **Enter** key. To change the decision, type the letter that appears in **( )**. See "Getting a Transcript" on page 2-16.

3. Each participant must agree to excuse you by typing **! excuse** *your username*, or you continue to receive conference comments on your display. For this example, each participant types:

   ```
   !excuse chr
   ```

   You continue to see the request for transcript information.

   The other participants see who has excused you. For example:

   ```
   jmc has excused chr
   ```

Getting a Transcript

As you withdraw from a conference, you confirm or change the transcript request made in the conference call by responding to a prompt.

The prompt offers you the opposite choice of the decision that was made during the conference call. For example, if the conference call was for an on-line conference (with transcript), you see:

```
Transcript (n)
```

- To change the original decision, type the letter in **( )**. In the above example, you cancel the transcript if you type the letter **n**.
- To accept the original decision, press the **Enter** key. In the example above, this confirms that you want a transcript.

If you request a transcript, it is sent to your **dead.letter** file, where it is identified by the conference name created by the system.

To read and handle the transcript, follow the instructions given in "Reading Your Mail" on page 2-20 and "Handling the Mail" on page 2-22.

Closing the Conference

Usually the conference leader stays until the end of the conference, but if the leader must leave, another participant can give the closing command.

1. All the other participants ask to be excused. As this is granted, the connection to the other users ends.

2. The last participant presses the **EOF** control key to end the conference.

Sending Mail (mail)

The **mail** command lets you send a note to a system user's mail box whether or not the user is logged in.

mail *username* Sends a note to the person indicated by the user name.

-t When added to the command, this flag adds the user name of each person to whom the note is sent.

If the system does not recognize one or more of the user names, the mail is stored in your **/usr/***username***/dead.letter** file so it can be edited and sent again.

How To Send Mail

1. Type **mail** *username* on the command line after the $ prompt.

   ```
   mail bjh
   ```

2. Press the **Enter** key.

   Unless the system prompt appears, the note can now be sent to `bjh's` mailbox.

3. Type the note.

4. End with a . (period) on a line by itself and press the **Enter** key.

   Instead of this last step, you can type **EOF**.

When the $ (command prompt) appears, you can do other work.

Several Mail Boxes

To send the same message to several mail boxes, list each *username*, as follows:

```
mail bjh chr doc
```

If you want recipients to know who else receives the note, add the **-t** flag to the command:

```
mail -t bjh chr doc
```

Each user name is added to the note.

To send notes to people on remote systems with the **mail** command, refer to Chapter 3.

Receiving Mail (mail)

You receive incoming mail in the **/usr/mail/***username* file.

When you receive mail, the following message displays:

```
[you have new mail]
```

If you receive mail while you are logged off, the message appears the next time you log in. If you are using the **mail** command when mail arrives, you are notified later, at the interval set for the mail check variable in your profile. (If the variable is set to **0**, the message appears each time you press the **Enter** key.)

If the system does not recognize your user name, the mail is sent to the sender's dead letter file, **/usr/***username***/dead.letter**.

Forwarding Your Mail

If you want your mail forwarded to a different user name, you can edit the **/usr/mail/***username* file so the first line reads:

```
Forward to username
```

Since this is a system file, you cannot edit it unless you have superuser authority.

Reading Your Mail

To read your mail, use the **mail** command with or without a flag. The flags let you decide in advance how to display or file incoming mail.

mail Displays your mail, last note first. Each note includes the sender's name, the date, and the time it was sent.

You are prompted by a **?** (question mark) to indicate how you want the mail handled. See "Handling the Mail" on page 2-22 for the choices you can make.

-f*newfile*
Saves your mail in the file you designate. Otherwise, your mail is saved in **/usr/***username***/mbox**, a file the system automatically sets up for you.

-p Displays your messages in read only state. The prompt for handling the mail is not displayed.

-r Displays your messages in the order they were received.

How To Read Your Mail:

1. Type the **mail** command after the $ prompt on the command line:

   ```
   mail
   ```

   Press the **Enter** key.

   The mail appears on the display, last note first. If you want the first note first, use the **-r** flag.

   ```
   mail -r
   ```

   Each item displays with a *postmark* containing the sender's name, the date, and the time the mail was sent. For example:

   ```
   From: bjh, June 19 12:04
   ```

2. Indicate after the **?** prompt how you want each note handled.

   See the next topic for a list of choices for handling mail. For example, if you want to see the next note, press the **Enter** key.

Handling the Mail

A **?** (question mark) command prompt appears after each note so you can indicate how you want that note handled. If the prompt does not appear, check to see if you chose the read-only option by using the **-p** flag when you gave the **mail** command.

To display the list of choices for handling the mail, type an ***** (asterisk).

Handling the Mail

1. Display the message you want to read. See the list of choices below. For example, to see the next message, press the **Enter** key.

2. Select an option to file or delete the message, by typing any specific information indicated by italics. See the list of choices below.

   For example, if you do not have time to handle all your mail and want to put the original mail file (with deletions restored) into your private mail box, **/usr/***username***/mbox**, type:

   ```
   s
   ```

   Since you do not want to specify another file, you do not need to type a file name.

3. Press the **Enter** key.

   When all the mail items are handled, the $ (command prompt) displays.

Options for Handling Mail

To Display the List of Choices: Type an ***** (asterisk).

To Display a Message: Press one of the following keys:

Enter	Displays the next message.
+ (plus)	Displays the next message.
- (minus)	Displays the previous message.
d	Deletes the displayed message and displays the next message.
p	Display current message again.

To Dispose of the Message: Press one of the following keys:

m *username*	Forwards the current message to another user, *username*.
q or Ctrl-D	Lets you quit reading mail. Returns the mail you have not deleted to the system file in which it was received, **/usr//mail/***username*, and exits from the **mail** command.
s *filename*	Saves the current message and its postmark in your private mail box, **/usr/***username***/mbox**, or in the specified file.
w *filename*	Saves the current message without its postmark in your private mail box, **/usr/***username***/mbox** or in the specified file.
x	Lets you exit from the **mail** command. Returns the original mail file (with deleted files restored) to the system file in which it

was received, **/usr/mail/***username*, and exits from **mail**.

!*command*	Runs shell *command* from **mail**.

If you deleted a message from the mail box by using the **d** (delete), **m** (move), **s** (save), or **w** (save without postmark) commands, and have not pressed **q** or **Ctrl-D** (quit), you can return the message to the mailbox by typing **x**. This returns the entire mail file to its original state.

See *AIX Operating System Commands Reference* for more information about the **mail** command.

Chapter 3. Creating a Remote Communication Facility with Base System Program Commands

CONTENTS

About This Chapter 3-3
Sending Mail to a Remote System (mail) 3-4
Connecting Your Local System to a Remote System (connect) 3-7
Providing Program Information 3-7
Setting Up a Stanza 3-9
Making a Connection to a Remote System 3-12
Making a Connection 3-13
Using Connect Subcommands 3-15
Running a Local Command 3-17
Copying Text Files to the Remote System 3-18
Keeping a Transcript File 3-20
Ending a connect Session 3-22

About This Chapter

This chapter describes how you can create a remote system communication facility using two commands that are an integral part of the AIX Base System Program. These commands are installed when you install AIX.

You can create a remote communication facility to:

- Send mail to people on remote systems, using the **mail** command
- Make a connection to a remote system, using the **connect** command.

The **mail** command is the same command used for local system communication, discussed in "Chapter 2. Creating a Local System Communication Facility."

The **connect** command discussed in this chapter is different from the ATE **connect** command discussed in "Chapter 5. Using Asynchronous Terminal Emulation (ATE)." We refer to the command in this chapter as **connect**, and the ATE command as ATE **connect**.

Sending Mail to a Remote System (mail)

You can send notes to a user's mail box on another system with the **mail** command if you know the name of the system and have a connection to it.

The format for the command is:

`mail [options]` *systemname*`!`*username*

mail Sends a note to the mail box of the person indicated by the destination system name and user name.

-t Option that adds the user name of each person to whom the note is sent

systemname
Name of the remote system

username Person to whom the mail is sent

How to Send Remote Mail

1. Type **mail** [options] *systemname!username*

   The following shows how to send mail to user `rumpole` on system `mack4`:

   ```
   mail mack4!rumpole
   ```

   No option is used in this example.

2. Press the **Enter** key.

3. Type the note.

4. End with a . on a line by itself, then press the **Enter** key.

   After the note is typed, you can use **Ctrl-D** to send the note instead of using this last step.

To use an intermediate system to route the mail, add an ! intermediate *systemname* before the destination system name.

```
mail aztec!mack4!rumpole
```

Additional Information

- Be sure any intermediate system is connected to your system and to the destination system.
- You can use any number of intermediate systems. Just list them in the same order as the physical connections.
- If the remote system does not recognize the user name, the mail is stored in your **/usr/***username***/dead.letter** file so it can be edited and sent again.
- Receiving, reading, and handling remote mail are tasks performed on the local system by the person to whom the mail

is sent. See "Receiving Mail (mail)" on page 2-20 for information on these tasks.

- Refer to *AIX Operating System Commands Reference* for more information about the **mail** command.

The rest of this chapter tells how to use the **connect** command to make a connection to a remote system, log into that system, and transfer text files from your system to the remote one.

Connecting Your Local System to a Remote System (connect)

You can establish a connection to a remote computer system with the **connect** command.

Once you are logged into the remote system, you can work on that system just as if you were directly connected. You can:

- Return (escape) to the local system prompt to run a command.
- Run commands on the remote system.
- Copy a text file to the remote system.
- Get a transcript of your session.
- Exit from the remote system.

Providing Program Information

Before you can establish a connection with a remote computer system, you must provide the **connect** program with information that:

- Defines the characteristics of the remote connection
- Defines the physical devices your system uses for the connection.

To do this, you must write two blocks of code, called stanzas, and add them to the **/usr/lib/INnet/connect.con** file.

Connect Stanza

This stanza describes the connection. It is wise to select a stanza name that identifies the system to which the connection is made, since you may need other stanzas to describe other connections.

Before you begin, collect the following information about the local and the remote system:

system name
Name by which the system is recognized.

login name Name that other systems use to call the system.

password Identifying word that protects login security.

address Telephone number that remote systems use to call the system.

No address is needed if the connection is a direct one via a cable or a modem eliminator. (In some files, a direct connection is called a permanent connection.) If the telephone number needs a prefix for dialing purposes, include the prefix in the address.

type of connection
May be direct or through a modem. Leave this blank if the connection is direct.

If the connection is made through a modem, both systems must use the same type of connection.

line speed May be 110, 300, 1200, 2400, or 9600 bits per second.

Must be the same for both systems.

parity May be even, odd, any, or none.

Must be the same for both systems.

Physical Device Stanza

The second stanza you must consider, placed at the end of the file, identifies characteristics of the physical devices used in the connection. A physical device stanza is needed when the connection is through an autodialer, such as a modem. The same characteristics must be used for all the connections described in the file.

For a modem, this stanza must include the:

- Name of the dialer
- Device
- Modem speed.

Setting Up a Stanza

A sample file, named **connect.con.t**, comes with the program.

The following procedures show how to edit the sample file to include a connect stanza for each connection you want, and a physical device stanza that describes the device characteristics. You need superuser authority to edit this file.

Setting Up a Connect Stanza

1. Copy the **/usr/INnet/connect.con.t** sample file into a file named **/usr/INnet/connect.con**.

2. Read the **connect.con** file into a text editor.

3. Find the stanza headed by this comment: `* connect stanza`. The sample stanza looks like this:

```
* connect stanza
remote:
     type = terminal
     use = modem
     address = "T5551234"
```

4. Change the stanza so it describes the name, type, use (connection device), and address of the remote system.

5. Find the stanza at the end of the file headed by this comment: `*stanza for information common to the above stanzas`. This is the stanza on physical device characteristics. The sample stanza looks like this:

```
*stanza for information common to the above stanzas
   modem:
     connect = name
     device = /dev/tty0

     speed = 1200
```

6. If you use an autodialer, rather than a direct connection cable, change the stanza so it describes the connection device (use), device name (connect), and line speed of the physical device used to make the connection.

7. Save the **connect.con** file.

For example, you may want to set up a local workstation to function as a terminal on remote SYSTEMB. If we assume the remote system is dialed on phone number 5553333 by a Hayes_1200 modem, and that the modem uses device file /dev/tty0 and a 1200 bits/second transmission speed, the stanzas would look like this:

```
* connect stanza
 systemb:
     type = terminal
     use = modem
     address = "T5553333"
* stanza for information common to the above stanzas
  modem:
     connect = Hayes_1200
     device = /dev/tty0

     speed = 1200
```

Additional Information

- The **connect** stanza name can be anything. However, it is recommended that you develop a naming strategy for **connect** stanzas, such as the name of the remote system.
- The T prefix of the address indicates that the address will be transmitted as a tone dialing sequence. Pulse (rotary) dialing is assumed if the T is deleted.
- The address must be enclosed in " " (double quotation marks).
- For detailed information on what you can define in a **connect.con** file, see *AIX Operating System Technical Reference.*
- To use the **connect** features, the information required by the **connect** program must be configured on your system. To see your system configuration, check the **connect.con** file or see your system administrator.

Making a Connection to a Remote System

To use the **connect** program, you must know the name of the **connect** stanza for the remote system.

The following sections describe how to start and stop a **connect** session, and how to use **connect** subcommands to transfer small files and keep a transcript of the session.

Making a Connection

Before your computer system can successfully initiate a **connect** session with a remote system:

- Your port must be disabled.
- The port on the remote system must be enabled.

For information on enabling and disabling ports, see Chapter 9, "Using Ports, Cables, and Modems" on page 9-1. Check with your system administrator and the system administrator at the remote system to ensure that the port on the remote system is enabled.

How to Make a Connection

1. Enter the **connect** command with the name of the **connect** stanza. For example:

   ```
   $ connect  systemb
   ```

2. The **connect** program displays a message telling you the device **connect** is using and the address of the remote terminal. For example:

   ```
   Using /dev/tty0...trying T8885359...
   ```

   Here `/dev/tty0` is the name of the device that **connect** is using to make the connection, and `T8885359` is the address of the remote device, in this case a telephone number.

   If **connect** succeeds in making the connection, it adds `connected` to the message.

   ```
   Using /dev/tty0...trying T8885359...connected
   ```

3. Once a connection is established, you can log into the remote system with a valid login name and password.

Additional Information

- If you are using a modem to link to the remote system, be sure the modem is switched on and connected to a phone line.
- Since the AIX Operating System distinguishes between lowercase and uppercase characters, be sure to find out exactly how the stanza name is spelled in the **connect.con** file.
- After the connection is made, the **connect** program displays a message describing the escape sequence to the login prompt of the remote system. For more information on escape sequences, see "Using Connect Subcommands" on page 3-15.
- If **connect** is not successful, it displays a brief message explaining the reason why the connection was not made. See *Messages Reference* for more information about these messages.
- If you see the message `connected` but you do not get a `login:` prompt, the connection is not complete. See your system administrator or refer to "Appendix B. Problem Determination."

Once you are logged in to the remote system, you can work on that system just as if you were directly connected. To exit the **connect** program and return to the local system:

- Escape to the local prompt (see "Using Connect Subcommands" on page 3-15).
- Quit the **connect** program (see "Ending a connect Session" on page 3-22).

Using Connect Subcommands

If you want to give subcommands to the **connect** program while you are connected to the remote system, you must first escape to the local **connect** prompt by using the **connect** escape sequence.

How to Escape to the Local Prompt

1. Press **Ctrl-V u Ctrl-M**.

   The **connect** program presents the `LOCAL:` prompt.

2. Enter a **connect** subcommand.

   The **connect** program performs the subcommand, presents the message `BACK TO REMOTE`, and returns you to the remote system.

3. Either enter another command for the remote system, or press **Enter** to get the remote system prompt (`$`).

Subcommands

The **connect** program subcommands you can use after the local system prompt follow:

! Signals to the **connect** program that a local system command follows. Use an **!** (exclamation point) in front of every local system command that you give at the `LOCAL:` prompt while you are in a **connect** session. If you do not do this, the **connect** program tries to run the command. You do not need an ! in front of **connect** subcommands given at the `LOCAL:` prompt. See "Running a Local Command" on page 3-17 for examples.

i*filename* Sends a local file to a remote file. For this subcommand to work, you must previously have used the **cat** command while the `BACK TO REMOTE` prompt appears to indicate where the file should go, then

escaped to the LOCAL: prompt to give this subcommand. See "Copying Text Files to the Remote System" on page 3-18 for the complete procedure.

t*filename* Transcribes a record of a connect session in the local system file you specify. To disable the transcript function, use a - (minus sign) after the **t**. To re-enable the transcript function, use a + (plus sign) after the **t**. See "Keeping a Transcript File" on page 3-20.

q Ends (quits) the remote session.

Additional Information

- Enter the escape key sequence by typing the following keys in the specified order:

  1. Press the **Ctrl** and **V** keys simultaneously, and then release them.
  2. Press the **u** key, then release the key.
  3. Press the **Ctrl** and **M** keys simultaneously, and then release them.

  Your system administrator can define a different escape key sequence. If the **Ctrl-V u Ctrl-M** does not work, ask your system administrator for the current escape sequence.

- The system administrator can also change the LOCAL: prompt string, so the prompt on your display screen may differ.

Running a Local Command

To run a command on the local system while you are in a **connect** session, use the ! subcommand to enter the command at the LOCAL: prompt.

How to Run a Local Command

1. Type !, followed by the command.

   For example, to display information about the files in the local current directory, type:

   ```
   !ls -l
   ```

2. Press the **Enter** key.

   An example of the information that appears is:

   ```
   total 2
   -rw-r--r-- 1 root root   80 Feb 123 14:54 testfile

   BACK TO REMOTE
   ```

Additional Information

- Be sure to type ! before the command that you want to run on the local system. Without the ! preceding the command, the **connect** program interprets what you enter as a command.
- If you do not enter ! before the command, the **connect** program usually generates a message to indicate that it cannot interpret and run the command.

Copying Text Files to the Remote System

You can copy a text file from the local to the remote system. The procedure uses the AIX **cat** command on the remote system, followed by the **i connect** subcommand at the `LOCAL:` prompt. The next box shows you how to do this.

Copying a Text File to the Remote System

1. While you are logged into the remote system, enter the **cat** command, followed by the name of the file where you want the local file copied. For example:

   ```
   BACK TO REMOTE
   cat > newfile
   ```

   This command tells the remote system to copy its standard input into the file `newfile`.

2. Press **Ctrl-V u Ctrl-M** to escape to the `LOCAL:` prompt.

3. Enter the **i** subcommand with the full path name of the text file you want to copy. For example, to copy `testfile`:

   ```
   LOCAL: itestfile
   ```

4. The following message appears:

   ```
   Including file :'testfile'
   ```

   The contents of `testfile` appears on the display screen and is transferred into `newfile` on the remote system.

5. Press **EOF** after transmission is complete. On the RT PC, use **Ctrl-D**.

Additional Information

- After you give the **cat** command, the remote system waits for input.
- Copying a large file with **connect** can tie up your system for a long time, because the file contents are displayed as the copy task is performed. You might prefer to use **uucp**, which runs as a background process.

Keeping a Transcript File

You can use the **t** (transcript) subcommand to keep a record of the remote **connect** session. This transcript is written to a local file that you specify.

The transcript file contains a record of the commands that you enter on the remote system and the remote system's response to those commands. It does not include the local commands that you issue at the `LOCAL:` prompt during the **connect** session.

The transcript function is enabled with the **t** subcommand. After the function is enabled, you can turn it off and on during the remote **connect** session. The transcript function for a remote **connect** session ends when the session is ended.

The following boxes show how to:

- Enable the transcript function during a remote **connect** session.
- Turn the transcript function off and on during the remote session.
- View the contents of the transcript file after the remote session ends.

How to Enable the Transcript Function

1. At the LOCAL: prompt, enter the **t** subcommand with the name of the file in which the record is to be stored:

   For example:

   ```
   LOCAL: tmyrecord
   ```

2. The system displays the following message:

   ```
   Transcription into file 'myrecord' enabled

   BACK TO REMOTE
   ```

3. The system begins keeping a record of the remote **connect** session in myrecord.

How to Disable the Transcript Function

Enter the following at the LOCAL: prompt:

```
t-
```

How to Re-Enable the Transcript Function

Enter the following at the LOCAL: prompt:

```
t+
```

If you want to view the contents of the transcript file, use the AIX **pg** command.

Viewing the Contents of the Transcript File

1. End the remote **connect** session.
2. Enter the following at the local AIX prompt:

```
$ pg myrecord
```

Ending a connect Session

Use the **q** subcommand to end a **connect** session. The **q** subcommand:

- Stops processes associated with the **connect** program
- Returns you to the command line in your working directory on the local system.

The following box shows how to end a **connect** session.

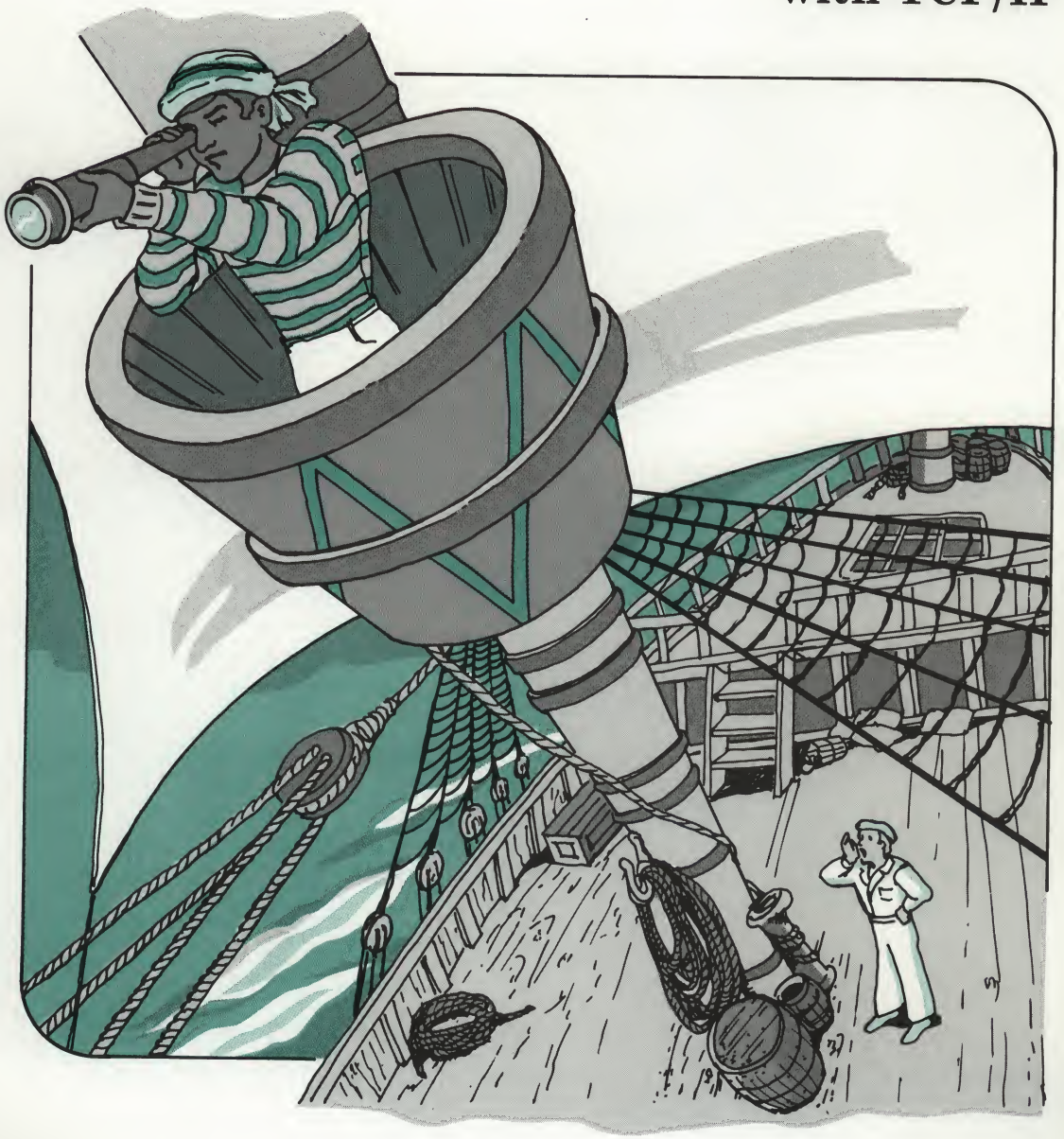
How to End a connect Session

1. Log off the remote system and escape to `LOCAL:` prompt.
2. Type `q` at the `LOCAL:` prompt.
3. Press the **Enter** key.
4. The system displays the message:

```
connection terminated.
```

5. The local AIX prompt ($) appears on your display screen.

Chapter 4. Using the Interface Program for Use with TCP/IP



CONTENTS

About This Chapter 4-3
Before You Begin 4-4
Overview of TCP/IP 4-4
Requesting Information about Users (finger) 4-5
Requesting Information about Remote Systems (ping) 4-7
Transferring Files (xftp) 4-8
Sending Mail (netmail) 4-14
Using a Remote Login (tn) 4-17
  Defining the Local and the Remote Consoles 4-17

About This Chapter

This chapter shows you how to communicate from one RT to another or to another system on a network, such as Ethernet[1] You do this with the Interface Program for use with TCP/IP (Transmission Control Protocol/Internet Protocol).

TCP/IP is on a diskette provided with AIX, and must be installed as a separate facility. The hardware and software needed for the network also must be installed, if not done previously.

An overview of the installation procedure for TCP/IP is in Appendix A. If you need additional information, see *Installing and Customizing the AIX Operating System* and *Interface Program for use with TCP/IP.*

---

1 Ethernet is a trademark of Xerox Corporation.

Before You Begin

If your system has not been customized for the network, follow the instructions in Appendix A of the manual *Interface Program for use with TCP/IP* after the installation of TCP/IP and the network is complete. Customization includes:

- Using the **devices** command to add the network device, configure ports for Telnet (the protocol that opens the connection to the system), and record the correct interrupt level.
- Customizing several files.
- Defining routes, if needed, using the **route** command.

Overview of TCP/IP

TCP/IP provides the commands you need to:

- Transfer files to and from another RT PC or other system
- Send mail to another system
- Log in remotely to another RT PC or other system.

This chapter discusses the basic commands you need to perform these tasks. For more information, including network management tasks not discussed here, see *Interface Program for use with TCP/IP*.

Requesting Information about Users (finger)

To request information about users on a specified system, use the **finger** command.

Format

```
finger [user]@systemname
```

finger Displays information about current users on the specified system. If you do not provide a user name, the **finger** command provides a list of all the current users. If a user name is specified, the following information is displayed:

- Login name
- Full name
- Terminal name and write status. An * indicates that write status is denied.
- Idle time
- Login time
- Office location, office phone, and home phone (if known)
- Contents of certain files in the **HOME** or **login** directory may be given.

user User name on the remote system.

@systemname
Name of a remote system.

Requesting User Information

To see a list of current users on a remote system:

1. Type `finger @`*systemname* after the **$** prompt on the command line.

   For example, to see a list of current users on remote system `chelsea`, type:

   ```
   finger @chelsea
   ```

2. Press the **Enter** key.

To see information about a specific user on a remote system:

1. Type `finger` *user* `@`*systemname* after the **$** prompt on the command line.

   For example, to see information about a user named `jones` on remote system `chelsea`, type:

   ```
   finger jones @chelsea
   ```

2. Press the **Enter** key.

Requesting Information about Remote Systems (ping)

To determine the status of the network and various remote systems, use the **ping** command.

Format

```
ping [option] systemname
```

ping	Determines the status of the network and other systems.
option	There are options available that are of interest to system programmers. See *Interface Program for use with TCP/IP*.
systemname	Name of remote system.

Transferring Files (xftp)

You can transfer files to and from another system with the **xftp** command. This is a two-step process:

1. The **xftp** command makes a connection to the other system.
2. Once the connection is made, subcommands that you give accomplish the file transfer.

See "Subcommands for xftp" on page 4-10.

Format

`xftp` *systemname*

xftp Makes a connection to another system so files can be transferred. This is the interface to the File Transfer Protocol (FTP).

systemname
Name of the system you want to reach. This may be another RT PC or another system to which you have a connection. The remote system is sometimes called a host.

When you see the `xftp>` prompt, give the subcommands that you need to make the file transfer.

Making the xftp Connection

1. Type **xftp** *systemname* after the **$** on the command line.

   If you want to reach a system named `host2`, type:

   ```
   xftp host2
   ```

   When the connection is made, you are notified and prompted for a login name: `Name (host2:`*local username*`)`.

2. To log into the remote system with your local system name, press the **Enter** key.

   For example, if you used `smith` on the local system, press the **Enter** key when you see: `Name (host2:smith)`.

   To log into the remote system with a different user name, type the name after the displayed information and press the **Enter** key. To login as `sam`, add the name as shown:

   ```
   Name (host2:smith) sam
   ```

3. If prompted, type a valid password, then press the **Enter** key. (The password does not appear on the screen.) The prompt for the previous example is:

   ```
   Password (host2:sam)
   ```

4. The prompt changes to `xftp>`. You now can enter any **xftp** subcommand. See the list of subcommands and the steps for transferring files that follow.

Subcommands for xftp

The following set of **xftp** subcommands lets you transfer files and perform related tasks, such as changing the type of file transfer, displaying information, and changing directory and filenames. For a complete list of subcommands, refer to the *Interface Program for use with TCP/IP*.

! (exclamation mark)
Invokes a shell on the local system so you can give a shell command.

append *localfile* **[***remotefile***]**
Appends a local file to a file on the remote system. If no remote file name is given, the local file name is used.

ascii Sets the file-transfer type to support network ASCII. This is the default. File transfer may be more efficient with binary-image transfer. Refer to the **binary** subcommand below.

binary Sets the file-transfer type to support binary-image transfer. This can be more efficient than an ASCII transfer, the default.

cd *directoryname*
Changes the working directory on the remote system to the directory you name.

delete *filename*
Deletes the file you name on the remote system.

dir [*directoryname***]**
Displays the contents of the directory you name on the remote system. If you omit a directory name, **xftp** displays the contents of the current directory.

get [*localfile***]** *remotefile*
Retrieves a file from the remote system and stores it on your local system. If no local file name is given, the remote file name is used.

help [*command*]
Displays a help message about the command you name. If you do not specify a command, **xftp** displays a list of known commands.

lcd [*directoryname*]
Changes the working directory on your system to the directory you name. If you do not specify a directory, **xftp** uses your **login** (HOME) directory.

ls [*directoryname*]
Displays an abbreviated list of the contents of the directory on the remote system. If you do not specify a remote directory (sometimes called a foreign directory), the contents of the current directory are listed.

mget (*remotefile*)
Retrieves multiple files from the remote system and stores them on your local system. The remote file names are used.

mput (*localfile*)
Sends multiple files from your local system to the remote system. The local file names are used.

put *localfile* [*remotefile*]
Stores a local file on the other system. If no remote file name is given, the local file name is used.

pwd Displays the name of the current directory on the remote system.

quit Ends the File Transfer Protocol session and exits the **xftp** program.

rename *fromname toname*
Changes a file name on the remote system.

How to Use Transfer Files

When the `xftp>` prompt appears, you are logged into the remote system and can transfer files or do other tasks related to file transfer.

1. Type the subcommand for file transfer or a related task, adding any required file or path name. Then press the **Enter** key.

2. Continue entering subcommands until all the work is finished.

   One possible sequence is given in the next box.

3. To exit **xftp**, enter the **quit** subcommand.

   ```
   quit
   ```

Before you transfer any files, you may want to issue other subcommands. The example in the box that follows shows a typical sequence that changes the type of file transfer from the default (ASCII), and puts a file into a specific directory on the remote system.

Sample Sequence for File Transfer

1. Change the file transfer type to binary by entering:

   ```
   binary
   ```

2. Check to see which working directory you are in on the remote system by entering the **pwd** command.

   ```
   pwd
   ```

3. Change the remote working directory by entering the **cd** subcommand. For example, to change to the `projects` directory, enter the following:

   ```
   cd projects
   ```

4. Transfer one file from the local to the remote system by entering the **put** subcommand. For example, to transfer the `/blue` file to `/u/sam/blue`, type:

   ```
   put /blue /u/sam/blue
   ```

Additional Information

File names must follow the conventions the **sh** command uses.

For more information, consult *Interface Program for use with TCP/IP*.

Sending Mail (netmail)

You can send mail to another system with which you have a connection, using the **netmail** command. The mail can be either a file or a note that you create.

Format

`netmail` [*mailfile*]
username@systemname

netmail Sends a mail file or note to another system.

mailfile The name of the file you want sent. If you omit a mail file name, the program assumes that you want to create a note.

username@systemname
The name of the user and the name of the system to which the mail is sent, joined by an @ (at) symbol.

How to Send a File Using Netmail

1. Type **netmail** [*mailfile*] *username@systemname* after the prompt on the command line. Then press the **Enter** key.

   To send a file named `march`, enter:

   ```
   netmail march smith@acct
   ```

If you want to send a note using **netmail**, you must create the note and then send it. How you do this depends upon whether the environment variable **EDITOR** is specified.

How to Send a Note Using Netmail

1. Type **netmail** *username@systemname* after the prompt on the command line. Then press the **Enter** key.

   To create a note for `smith`, type:

   ```
   netmail smith@acct
   ```

   If the environnment variable **EDITOR** is specified, an editor appears on the display. If the variable is not specified, the display does not change, and the program takes the input you type on the command line.

2. If an editor appears:

   a. Type the information as directed for that editor.

   b. Exit from the editor.

      The note is sent automatically when you exit, and the file in your current directory is deleted.

   If an editor does not appear:

   a. Type the text of the note on the command line.

      Press the **Enter** key after each line of text.

   b. When the text is complete, press **Ctrl-D** to send the note.

Additional Information

- If you do not give a mail file name, the program presents the editor specified in the environment variable **EDITOR**. If no environment variable **EDITOR** is specified, the default value `standard input` is in effect. This means that the text you type on the command line will be sent to a buffer for distribution.

- The **netmail** command provides an interface to the Simple Mail Transfer Protocol (SMTP), and the **smtp** program completes the transfer.

- To read the mail, use the **mail** command in the AIX Base System Program. Refer to "Receiving Mail (mail)" on page 2-20. If mail cannot be delivered, it will be returned to you, rather than sent to your dead letter file **usr/***username***/dead.letter**. Otherwise, you receive, read, and handle mail as for the **mail** command.

Using a Remote Login (tn)

You can log into another system with which you have a connection, using the **tn** command. The **tn** command implements the Telnet protocol, which opens a connection to the system.

Using a remote login is a two step process:

1. The **tn** command logs into a remote system.
2. Subcommands that you give manage the remote session.

Defining the Local and the Remote Consoles

If you plan to communicate from an RT PC console at your local system to another RT PC console at the remote system, skip this section. You are now ready to give the **tn** command. See "Format" on page 4-18.

If you want to communicate with a system that does not have an RT PC console, you must define the local and the remote console in a language that both understand. To do this:

1. Change the environment variable **EMULATE** by typing `EMULATE=`*value*`;export EMULATE` after the system prompt.

   If **EMULATE** = `vt100`, the **tn** program emulates a DEC VT100[2] terminal. If **EMULATE** is not defined or has a value other than `vt100`, you see an RT PC display.

2. If **EMULATE** = `vt100`, be sure the value of **TERM** in the remote login connection also is `vt100`.

   You can check this by using the **env** command after the connection is open, and change it by typing `TERM=vt100` after the system prompt.

---

2 DEC and VT100 are trademarks of Digital Equipment Corporation.

You are now ready to give the **tn** command.

Format

`tn [-p` *port*`]` *systemname*

tn Provides an interface to the Telnet protocol so you can log into the system you specify.

-p *port* Opens a connection to the specified port instead of to the Telnet default port.

systemname
The name of the system where you want to log in. If you omit a systemname (sometimes called a host), the **tn** program prompts you for one.

How to Log Into and Use a Remote System

1. Type **tn** *systemname* after the prompt on the command line. For example, if you want a connection to *syst2*, type:

   ```
   tn syst2
   ```

   If you do not give a system name, you are prompted to do so.

   When the command is accepted, several lines of message text appear on the display, ending with the login prompt.

2. When prompted, type your login name and press the **Enter** key.

   When the system prompt appears, you are logged into the remote system and can transmit text or do related tasks, using the **tn** subcommands.

3. Type the text you want to send.

   If you want to receive status or help information, or perform some related task, use the correct subcommand on the next page. Before entering each subcommand, type **Ctrl-T**.

4. To close the connection and exit the program, use the **Ctrl-R** key combination.

Subcommands for Remote Login

The following list is a subset of the **tn** subcommands. For a complete list, refer to *Interface Program for use with TCP/IP*.

Before entering each subcommand, type **Ctrl-T**. **Ctrl-T** is an escape sequence that tells the program that non-text information follows. Otherwise, the program would interpret subcommands as text.

? (questionmark)
Displays a help message for available Telnet commands.

a Asks if the Telnet Protocol is still connected.

b Sends the Telnet **break** command in urgent mode.

c Closes the Telnet connection and exits.

p Suspends **tn** and returns to the shell. To return to **tn**, press the **Ctrl-D** control key.

q Quits Telnet immediately, without waiting for the connection to close. Use **q** if the remote system terminates the connection abnormally.

s Displays the status of the connection.

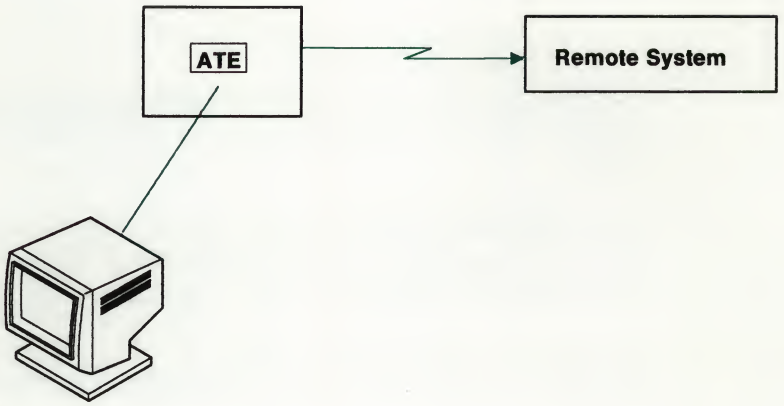
Chapter 5. Using Asynchronous Terminal Emulation (ATE)

CONTENTS

About This Chapter 5-3
Overview of Asynchronous Terminal Emulation (ATE) Tasks 5-4
  Unconnected Main Menu 5-5
  Connected Main Menu 5-7
Giving Commands 5-8
  Using Control Keys 5-9
  Functions of Control Keys 5-9
Prerequisite Tasks 5-12
Starting ATE 5-14
Making a Connection (ATE connect) 5-14
  Manual Dialing 5-15
  Automatic Dialing 5-17
  Direct Connection 5-19
Displaying a Dialing Directory (directory) 5-21
  Selecting a Telephone Number from a Directory 5-22
Creating a Dialing Directory File 5-23
  Description of Directory Name 5-23
  Dialing Directory File Format 5-26
Sending a File (send) 5-28
Receiving a File (receive) 5-30
Interrupting a Session (break) 5-32
Terminating a Session (terminate) 5-32
Getting Help (help) 5-33
Performing an Operating System Command (perform) 5-35
Leaving ATE (quit) 5-36

About This Chapter

This chapter shows you how to communicate with a remote computer system by emulating a remote terminal. The illustration below provides a general overview:



The tasks described here are the tasks you use to establish a connection and log into a terminal on another system (such as a data base service). Then, you can give commands, use files, and run programs on that system.

Although customization and configuration are prerequisites for using Asynchronous Terminal Emulation (ATE), some tasks (like giving port commands) require superuser authority, and others require some data processing experience. Therefore, an overview of the prerequisite tasks is provided in this chapter so that you can check to be sure the system is ready to use. The detailed discussion occurs in Chapter 6, "Customizing Asynchronous Terminal Emulation (ATE)." Information on configuring ports is discussed in Chapter 9, "Using Ports, Cables, and Modems."

To install ATE, use the ATE diskette that comes with AIX. An overview of the installation procedure for ATE is in Appendix A. For more information, see *Installing and Customizing the AIX Operating System.*

Overview of Asynchronous Terminal Emulation (ATE) Tasks

Before you can use ATE, you must learn the basic ATE tasks and understand the commands on the main menus that perform the tasks.

From the Unconnected Main Menu, which appears when you start the ATE program, you can establish a connection to another terminal two different ways:

- Make a direct connection.
- Use a dialing directory and make a telephone connection.

You can also check to be sure the settings for your local system and for the connection are correct, and change the settings for the current session, if necessary.

Tasks related to establishing a connection and changing settings are described in "Unconnected Main Menu" on page 5-5.

Once a connection is made, you can display the Connected Main Menu and select commands to:

- Send a file.
- Receive a file.
- Interrupt a session with a break signal.
- Terminate a connection.

These tasks are described in "Connected Main Menu" on page 5-7.

As you perform the above tasks, you may need to:

- Get help.
- Perform an operating system command.
- Leave (quit) ATE.

These last three tasks start from commands found on both of the main menus.

Next, we provide an overview of the main menus, and discuss how to give menu commands and use control keys.

Unconnected Main Menu

The Unconnected Main Menu displays any time you use the **ate** command, provided that Asynchronous Terminal Emulation is installed.

```
Node: Evelyn            UNCONNECTED MAIN MENU

     COMMAND        DESCRIPTION

     Connect        Make a connection
     Directory      Display a dialing directory

     Help           Get help and instructions
     Modify         Modify local settings
     Alter          Alter connection settings
     Perform        Perform an Operating System Command
     Quit           Quit the program

     The following keys may be used during a connection:
         ctrl b  start or stop recording display output
         ctrl v  display main menu to issue a command
     Use ctrl r  to return to a previous screen at any time

Type the first letter of the command and press Enter.
>
```

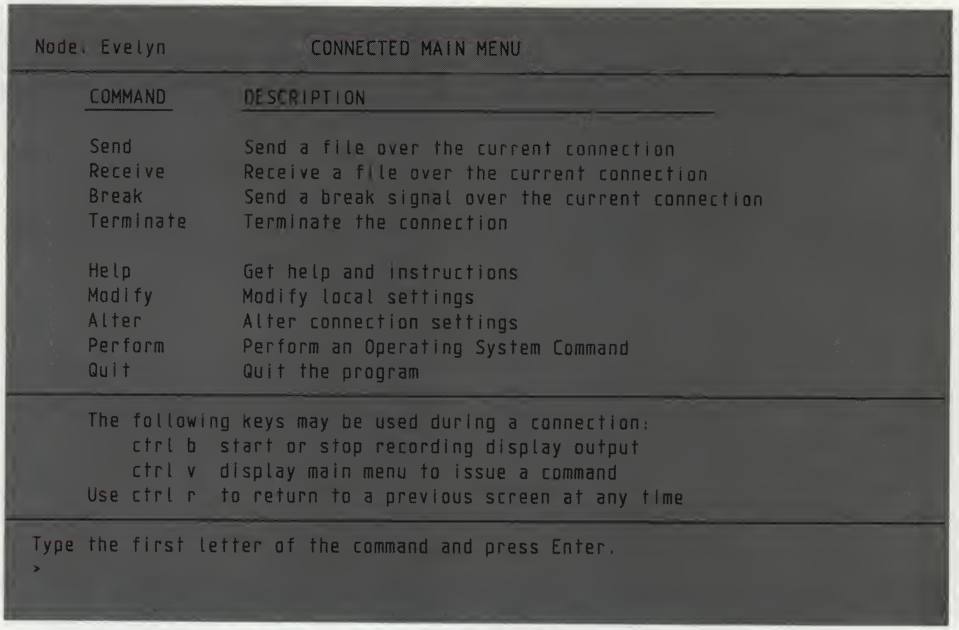
With the Unconnected Main Menu you can:

- Make an asynchronous connection to a remote computer.
- Display and use a phone directory to make a connection.
- Request help information.
- Request the Modify Menu to change variables that pertain to your local terminal (for the current session).
- Request the Alter Menu to change variables that affect data transmission (for the current session).
- Run an operating system command (some, like port commands, require superuser authority).
- Stop using Asynchronous Terminal Emulation.

To view the Connected Main Menu, give the **connect** command from the Unconnected Main Menu, then use the **Ctrl-V** control key.

Connected Main Menu

The Connected Main Menu contains the commands you need to transmit files. The Connected Main Menu looks like this:



With the Connected Main Menu you can:

- Send a file to a remote system.
- Receive a file from a remote system.
- Interrupt a connection with a break signal.
- Terminate a connection.
- Request help information.

- Request the Modify Menu to change variables that pertain to your local terminal (for the current session).
- Request the Alter Menu to change variables that affect data transmission (for the current session).
- Run an operating system command (some, like port commands, require superuser authority).
- Stop using Asynchronous Terminal Emulation.

Giving Commands

To give a command, type the first letter of the command after the > (prompt) at the bottom of the menu and press the **Enter** key. Notice that the menu prompt is a different symbol than the $ (system prompt) mentioned earlier.

For example, to display a dialing directory, type the first letter of the **directory** command:

```
d
```

Press the **Enter** key.

Do *not* try to enter values by using cursor movement keys to mark the command in the menu. The system interprets the cursor movement keys as typed input and fails to recognize the command. If you use cursor movement keys, the following error message appears:

```
062-003  The 'command-name' command is not valid.
         Enter the first letter of a command from
         the list on the menu.
```

If you see the above message, press the **Enter** key to redisplay the menu. Then try the command again.

Using Control Keys

Three control keys appear on both the Unconnected and the Connected Main Menus. These control keys do not function until a connection is made to a remote system by giving the **connect** command.

To use a control-key combination, hold the **Ctrl** key while you press the named letter-key.

Functions of Control Keys

The functions of the control keys follow:

Ctrl-B Starts or stops saving data displayed on your screen during a connection. This control key has a switch or toggle effect. The first time you use **Ctrl-B** you save data. The next time you use it, you stop saving data. This key combination does not function while files are being transferred.

Data is saved in the default file, unless you request a specific file. Changing the default file is discussed in Chapter 6, "Customizing Asynchronous Terminal Emulation (ATE)" on page 6-1.

If you press **Ctrl-B** when there is no connection, you receive an error message.

```
062-003  The 'command-name' command is not valid.
         Enter the first letter of a command from
         the list on the menu.
```

Since control characters do not normally display, the command name may be blank.

Ctrl-V Displays the Connected Main Menu so you can issue a command. Use this control key to display the Connected Main Menu after the **connect** command makes a connection.

If you press **Ctrl-V** from the Unconnected Main Menu and then press **Enter**, you receive message 062-003 (above).

Ctrl-R This sequence returns you to the previously displayed screen. The actual screen you see depends on the screen in use when the key combination was pressed.

You can use **Ctrl-R** to stop a file transfer in progress.

From the Unconnected State

Where You Are:	When You Use Ctrl-R:
Unconnected Main Menu	Ignored
Directory Menu	Unconnected Main Menu returns.
Help Menu	Unconnected Main Menu returns.
Modify Menu	Unconnected Main Menu returns.
Alter Menu	Unconnected Main Menu returns.
During a perform	Displays the message: Press any key
During a quit	Ignored
Making a connection	Terminates the connection. The menu you were using returns: Unconnected Main Menu or the Directory Menu.

From the Connected State

Where You Are:	When You Use Ctrl-R:
Connection Screen	Displays the message: `Disconnect (y/n)?` If you choose yes, either the Unconnected Main Menu or the Directory Menu appears, depending on which one you used to make the connection. If you choose no, the connection screen returns.
Connected Main Menu	Connection screen returns
Help Menu	Connection screen returns
Modify Menu	Connection screen returns
Alter Menu	Connection screen returns
During a perform	Displays the message: `Press any key`
During a quit	Ignored
During a terminate	Ignored
During a break	Ignored
During a send	Connection screen returns
During a receive	Connection screen returns

Prerequisite Tasks

Before you use the ATE program, you should check the settings of the local system, the connection, and the ports. You must know the following:

- Characteristics of the remote port
- Characteristics of the local port
- Settings for your local system
- Settings needed to connect to a remote system.

At the remote system, as a minimum you must know the transmission rate, the file transfer protocol, the bit length (per character), the parity, and which remote port is ready to receive calls. Someone at the remote system can check these and other characteristics for you so you can make your local port compatible.

At the local port, as a minimum you must be sure that the transmission rate, the file transfer protocol, the parity, and the bit length are the same as those of the remote system, and know which local call-out port is configured for the task you need.

You can check and change the settings of your local system for the current session by using the **modify** command. These settings include the following:

- Naming the file that captures incoming data
- Adding linefeeds at the end of each line of incoming data
- Setting echo mode
- Emulating a DEC VT100
- Writing incoming data to a capture file
- Establishing a transmitter on/off signal (Xon/Xoff).

When you leave ATE, the settings for the current session disappear, and the values in the default file are in effect the next time you start ATE. See "Modifying Local Settings (modify)" on page 6-4 for more information.

You can check and change the connection settings for the current session by using the **alter** command. These settings include the following:

- Bit length per character
- Number of stop bits
- Parity
- Transmission rate
- Device name of port
- Modem dialing prefix
- Modem dialing suffix
- Wait before redialing
- Number of redial attempts
- File transfer protocol
- Pacing character or integer

When you leave ATE, the settings for the current session disappear, and the values in the default file are in effect. See "Altering Connection Settings (alter)" on page 6-9 for more information.

If a port is ready to receive calls, it is ***enabled***. If a port is ready to call out, it is ***disabled***. The port configuration can be changed with the **devices** command. Remember that changing the port configuration requires superuser authority. See Chapter 9, "Using Ports, Cables, and Modems" on page 9-1 for more information.

Starting ATE

To start the ATE program, type **ate** after the $ (system prompt) and press the **Enter** key:

```
$ate
```

The **ate** command displays the first of two main menus, the Unconnected Main Menu, from which you establish a connection to another terminal. See "Unconnected Main Menu" on page 5-5.

Making a Connection (ATE connect)

Before you begin, review the list of prerequisite tasks and make sure that the configuration is complete. Remember that checking and changing the port configuration requires superuser authority, and that if you leave ATE after you change local settings or connection settings, the changes disappear.

The ATE **connect** command makes a connection between your terminal and a remote system with the method you select:

- Manual dialing
- Automatic dialing
- Direct connection.

The **connect** command is given from the Unconnected Main Menu, which appears when you start the program.

Manual Dialing

Manual dialing means that you dial the number yourself over a telephone line.

How to Dial Manually

1. Be sure all the prerequisite tasks are complete. See "Prerequisite Tasks" on page 5-12.

2. Invoke ATE by typing the following on the command line and pressing the **Enter** key:

   ```
   ate
   ```

3. If you do not need to change the current settings, skip this step. To change settings for the current session, when the Unconnected Main Menu appears, use the **modify** or the **alter** command. See "Modifying Local Settings (modify)" on page 6-4 and "Altering Connection Settings (alter)" on page 6-9 for instructions.

4. To make the connection, type the letter **c** on the command line.

   ```
   >c
   ```

5. Press the **Enter** key.

6. When the following prompt appears, press the **Enter** key to signal that you want to dial manually:

   ```
   Type the phone number of the connection for auto
   dialing, or the name of the port for direct connect,
   and press Enter.  To manually dial a number, just
   press Enter.  To redial the last number (0),
   type 'r' and press Enter.
   >
   ```

7. Dial the telephone number when message 062-007 appears.

Additional Information

- The prompt for dialing the telephone number appears as message 062-007. You can read more about this message in *IBM RT PC Messages Reference.*

  ```
  062-007  Please dial a telephone number to make the
           requested connection.
  ```

- If you hear a busy signal and want to end the connection request, press **Ctrl-R**.

- No connection is established if the line is busy, or if the party doesn't answer, or if you used an unrecognized number. The following message appears after 45 seconds:

  ```
  062-009  The 'connect' command cannot complete because
           the line was busy, or the modem did not detect
           a carrier signal.  Make sure the number is
           correct and try again, or try the same
           number later.
  ```

  Check the number and try again.

- Message 062-009 also appears if you use the **Ctrl-R** control keys. This stops a connection from being established. If you pressed **Ctrl-R** by mistake, try again.

Automatic Dialing

Automatic dialing means that you use a modem to dial a number over a telephone line.

How to Dial Automatically

1. Be sure all the prerequisite tasks are complete. See "Prerequisite Tasks" on page 5-12.

2. Invoke ATE by typing the following on the command line and pressing the **Enter** key:

   ```
   ate
   ```

3. If you do not need to change the current settings, skip this step.

   When the Unconnected Main Menu appears, use the **modify** or the **alter** command to change settings for the current session. See "Modifying Local Settings (modify)" on page 6-4 and "Altering Connection Settings (alter)" on page 6-9 for instructions.

4. To make the connection, type **c** *telephonenumber* and [*portname*] on the command line.

   ```
   >c 5551111 tty6
   ```

5. Press the **Enter** key.

Additional Information

- Your modem instruction book should tell you how how to format the number, perhaps by including spaces or dashes.

- Although the example above contains a port name, the [ ] (brackets) in the instruction indicate that a port name is optional. For information on port names, see Chapter 9, "Using Ports, Cables, and Modems" on page 9-1.

- If you give the ATE **connect** command without a telephone number or port name, the following prompt appears:

```
Type the phone number of the connection for auto
dialing, or the name of the port for direct connect,
and press Enter.  To manually dial a number, just
press Enter.  To redial the last number (0),
type 'r' and press Enter.
>
```

Type the telephone number to dial automatically:

```
>5551111
```

Press the **Enter** key.

Direct Connection

You can make a direct connection if you have a link to another system.

How to Make a Direct Connection

1. Be sure all the prerequisite tasks are complete. See "Prerequisite Tasks" on page 5-12.

2. Invoke ATE by typing the following on the command line and pressing the **Enter** key:

   ```
   ate
   ```

3. If you do not need to change the current settings, skip this step.

   When the Unconnected Main Menu appears, use the **modify** or the **alter** command to change settings for the current session. See "Modifying Local Settings (modify)" on page 6-4 and "Altering Connection Settings (alter)" on page 6-9 for instructions.

4. To make the connection, type **c** *portname* on the command line.

   ```
   >c tty6
   ```

5. Press the **Enter** key twice.

   The login prompt appears.

Additional Information

- Before you can make a connection to another computer, you must disable your port so that no one else can log into your system while you are trying to make a connection. To do this, someone with superuser authority must use the **pdisable** command.

  If you do not disable your system login, an error message appears.

  You must also be certain that the port of the other computer is ready to receive calls.

- If you give the ATE **connect** command without typing the port name, the following prompt appears:

  ```
  Type the phone number of the connection for auto
  dialing, or the name of the port for direct
  connect, and press Enter.  To manually dial a
  number, just press Enter.  To redial the last
  number (0), type 'r' and press Enter.
  >
  ```

  Type the port name on the command line and press the **Enter** key.

Displaying a Dialing Directory (directory)

The **directory** command displays a dialing directory. You can establish a connection to a remote computer by selecting one of the directory entries from the displayed directory. The information in this section tells you how to do this.

If you need to create a dialing directory, refer to "Creating a Dialing Directory File" on page 5-23.

The **directory** command must be given from the Unconnected Main Directory.

How to Display a Dialing Directory

1. Type **d** *filename*

   ```
   >d bulletin_bds
   ```

2. Press the **Enter** key.

Additional Information

- If you gave the **directory** command without typing the *filename*, the following prompt appears:

  ```
  Type the file name of the directory you want to
  display and press Enter.  To use the current
  directory (/usr/lib/dir), just press Enter.
  >
  ```

  If you see this prompt, type the name of the directory you want to display, unless the file name is inside the **( )**. Press the **Enter** key.

  To use the current directory (the one displayed in parentheses), just press the **Enter** key.

- Asynchronous Terminal Emulation comes with a sample dialing directory called **/usr/lib/dir**. The first time you use the **directory** command the sample directory appears. After that, the last directory you used becomes the current directory.

Selecting a Telephone Number from a Directory

When a directory is displayed, a prompt appears, asking which entry you want to select.

```
#        NAME            TELEPHONE (first digits)  RATE  LEN STOP PAR ECHO LF's

 0 CompuAid              555-0000                  1200   7    1   2    0    0
 1 Stock_Info            555-1111                  1200   7    1   2    0    0
 2 Info_Index            555-2222                  1200   7    1   2    0    0
 3 Electronic_Mail       555-3333                  1200   8    1   0    0    1
 4 The_Origin            555-4444                  1200   7    1   2    0    0
 6 John's_Extension      8,1111                    1200   8    1   0    0    0
 7 LD_Info               111,555-1212              1200   8    1   0    0    0
 8 Low_Cost_LD           555-5555,666666,800-555-77 1200  8    1   0    0    0
 9 Joe's_Pizza           9,555-8888                1200   8    1   0    0    0
10 bulletin_board        555-9999                   300   8    1   0    0    0
11 The_Lab               8,2222                    9600   8    1   0    0    0

Enter directory # or e (Exit)
>
```

1. Type the number of the entry you want (0 through 19).
2. Press the **Enter** key.

Creating a Dialing Directory File

This section tells you how to create a dialing directory to use with the **directory** command. For information on how to use the directory that you create, see "Displaying a Dialing Directory (directory)" on page 5-21.

A dialing directory file is similar to a page in a telephone book. Each file contains telephone numbers for remote systems you can call with Asynchronous Terminal Emulation.

Description of Directory Name

To help you create dialing directory files, Asynchronous Terminal Emulation comes with a sample dialing directory **/usr/lib/dir**. See "Dialing Directory File Format" on page 5-26.

directory Name of a dialing directory file.

Options: Any string of 40 characters or less.

Default: **/usr/lib/dir**.

Creating a Directory from the Model

1. Select a name for your dialing directory file that makes your new file unique. This must be a string of 40 characters or less.

   The file can be placed in any directory that has write permission.

2. Copy the sample directory to your new directory file, using the **cp** command.

3. Using your favorite AIX text editor, replace the sample entries with those you want in your directory, completing the information in each of the eight fields. (See "Dialing Directory File Format" on page 5-26.)

4. Save the new file.

Additional Information

- If you need to give a directory write permission, use the **chmod** command.
- See *AIX Operating System Commands Reference* for instructions on using the **cp** and the **chmod** commands.
- Limit the file to 20 entries, numbered from 0 through 19. If you add more than 20 entries, the ATE program ignores them.
- If you are using INed[1], refer to *INed* for help in creating files.

An alternate way to create a new directory follows:

1 INed is a trademark of INTERACTIVE Systems Corporation.

Creating a Directory without the Model

1. Create a new file with any AIX text editor.
2. Type each telephone number on a separate line.

   Each entry must include the eight fields described in the next section, with at least one space between each field. Do not exceed the maximum number of characters in each field.

   Two sample entries follow:

```
CompuAid        555-0000 1200 7 1 2 0 0
Stock_Info      555-1111 1200 7 1 2 0 0
```

3. Save the file.

Dialing Directory File Format

The sample directory you copy to model looks like this:

```
CompuAid            555-0000                          1200 7 1 2 0 0
Stock_Info          555-1111                          1200 7 1 2 0 0
Info_Index          555-2222                          1200 7 1 2 0 0
Electronic_Mail     555-3333                          1200 8 1 0 0 1
The_Origin          555-4444                          1200 7 1 2 0 0
John's_extension    8,1111                            1200 8 1 0 0 0
LD_Info             111,555-1212                      1200 8 1 0 0 0
Low_Cost_LD         555-5555,666666,800-555-7777      1200 8 1 0 0 0
Joe's_Pizza         9,555-8888                        1200 8 1 0 0 0
bulletin_board      555-9999                           300 8 1 0 0 0
The_Lab             8,2222                            9600 8 1 0 0 0
```

A description of each field follows:

name Name that identifies a telephone number.

Options: Any combination of 20 characters or less. Use the _ (underscore) instead of a blank between words in a name. An example is `data_bank`.

number Telephone number to be dialed.

Options: A telephone number of 40 digits or less.

Consult your modem user's guide for a list of acceptable digits and characters. For example, if you must dial 9 to access an outside line, include a 9 and a , (comma) before the telephone number.

`9,5551111.`

Only the first 26 characters (including digits, commas, and dashes) display in the Directory Menu.

rate Transmission or bit rate (bits per second). Determines the number of characters transmitted per second. Select a bit rate that is compatible with the capability of the communication line you are using.

Options: 50, 75, 110, 134, 150, 300, 600, 1200, 1800, 2400, 4800, 9600, or 19200.

length Number of bits that make up a character.

Options: 7 or 8.

stop Stop bits that signal the end of a character.

Options: 1 or 2.

parity Checks whether a character was successfully transmitted to or from a remote system.

For example, if you select even parity, when the number of 1 bits in the character is odd, the parity bit is turned on to make an even number of 1 bits.

Options: 0 = none, 1 = odd, 2 = even.

echo Determines whether typed characters display locally.

Options: 0 = off (no echo), 1 = on (echo).

linefeed Adds a linefeed character at the end of each line of incoming data from a remote system.

The linefeed character is similar in function to the carriage return and newline characters.

Options: 0 = off (no linefeed), 1 = on.

Sending a File (send)

The **send** command on the Connected Main Menu sends a file from your local system to a remote system, if a connection is established. Before you can send the file, you must prepare the remote system to receive it, as follows:

Preparing a Remote System to Receive a File

To send a file to another RT PC using **xmodem** protocol:

1. Make sure that your local system is disabled and the RT PC remote system is enabled. Someone with superuser authority must do this.
2. Make a connection to the other RT PC using the **connect** command:

   `c` *devicename*

   Press the **Enter** key.
3. When the login prompt appears, type:

   loginname

   Press the **Enter** key.
4. When the password prompt appears, type the password (if any) and press the **Enter** key. If there is no password, just press the **Enter** key.
5. Give the **xmodem** command on the other system by typing:

   `xmodem -r` *filename*

   Press the **Enter** key.

After you have set up the remote computer to receive the file, you must do the following:

How to Send a File to a Remote System

1. Press **Ctrl-V** to display the Connected Main Menu.
2. Type **s** *filename* on the command line, using the name of the file you want to send.

   For example, to send a file named `myfile`, type:

   ```
   >s myfile
   ```

3. Press the **Enter** key.

Additional Information

If you give the **send** command without typing the *filename*, the following prompt appears:

```
Type the name of the file you wish to send and
press Enter.  To use the last file name (   ),
just press Enter.
>
```

If you see this prompt, type the name of the file you want to send, unless the file name is inside the **( )**. Press the **Enter** key.

Receiving a File (receive)

The **receive** command stores incoming data from a remote system in the file that you designate. This command must be given from the Connected Main Menu.

Preparing a Remote System to Send a File

To receive a file from another RT PC using **xmodem** protocol:

1. Make sure that your local port is disabled and that the RT PC remote port is enabled. A superuser must do this.
2. Make a connection to the other RT PC using the **connect** command:

   `c` *devicename*

3. When the login prompt appears, type:

   loginname

   Press the **Enter** key.

4. When the password prompt appears, type the password (if any) and press the **Enter** key. If there is no password, just press the **Enter** key.
5. Give the **xmodem** command on the other system by typing:

   `xmodem -s` *filename*

   Press the **Enter** key.

How to Receive a File from a Remote System

1. Press **Ctrl-V** to display the Connected Main Menu.
2. Type **r** *filename* on the command line.

   For example, to store data in a file called `infile`, type:

   ```
   >r infile
   ```

3. Press the **Enter** key.

Additional Information

If you give the **receive** command without typing the *filename*, the following prompt appears:

```
Type the name of the file you wish to store the received
data in and press Enter.  To use the last file name (   ),
just press Enter.
>
```

If you see this prompt, type the name of the file in which you want to store the incoming data, unless the file name is inside the **( )**. Press the **Enter** key.

To use the file name displayed in parentheses, just press the **Enter** key.

Interrupting a Session (break)

The **break** command sends a break signal to the remote system to which your terminal is connected. The break signal interrupts current activity on the remote computer.

Warning: THIS SIGNAL MAY DISCONNECT THE CURRENT SESSION. YOU MAY LOSE DATA.

This command must be given from the Connected Main Menu.

How to Interrupt a Session

1. Type **b** on the command line.

   ```
   >b
   ```

2. Press the **Enter** key.

Terminating a Session (terminate)

The **terminate** command stops a **connect** session and returns you to the Unconnected Main Menu. This command must be given from the Connected Main Menu.

How to End a Session

1. Type **t** on the command line.

   ```
   >t
   ```

2. Press the **Enter** key.

Getting Help (help)

The **help** command displays a description of each command for which you request help, and provides instructions for using the command.

You may request **help** from either the Unconnected Main Menu or the Connected Main Menu for all the commands that appear on that menu. Help may be requested for several commands at the same time.

To access the general help screen, type **h** and press the **Enter** key.

How to View a Help Message

1. Type **h** [*commandinitial*] on the command line.

   For example, to receive help on the **receive** and **terminate** commands, type:

   ```
   >h r t
   ```

2. Press the **Enter** key.

Additional Information

Help information for each command displays individually, in the order requested. As you finish reading each help message, press the **Enter** key to view the next command description.

The [ ], (brackets) indicate that you may type as many command initials as you wish. Valid command initials are:

c connect

d directory

s	send
r	receive
b	break
t	terminate
m	modify
a	alter
p	perform
q	quit

Performing an Operating System Command (perform)

The **perform** command lets you run an AIX shell command from Asynchronous Terminal Emulation.

You can give the **perform** command from either the Unconnected Main Menu or the Connected Main Menu.

How to Run a Shell Command

1. Type **p** *shellcommand*

   For example, to display the data in `cheerfile` using the **cat** shell command, type:

   ```
   >p cat cheerfile
   ```

2. Press the **Enter** key.

Additional Information

If you give the **perform** command without typing the name of the shell command, the following prompt appears:

```
Type an operating system command and press Enter.
>
```

If you see this prompt, type the name of the command you want to run. Press the **Enter** key.

Note: For information on the **cat** command, see *AIX Operating System Commands Reference.*

Leaving ATE (quit)

The **quit** command exits the Asynchronous Terminal Emulation program and returns you to the operating system.

How to Leave the ATE Program

1. Type **q** on the command line.

   ```
   >q
   ```

2. Press the **Enter** key.

The system prompt appears, indicating that you may give any shell command:

```
$
```

Chapter 6. Customizing Asynchronous Terminal Emulation (ATE)

CONTENTS

About This Chapter 6-3
Before You Begin 6-4
Modifying Local Settings (modify) 6-4
  Description of Features 6-7
Altering Connection Settings (alter) 6-9
  Description of Characteristics 6-12
Changing (Remapping) the Control Keys 6-15
  ASCII Control Characters 6-16
Changing Your Default File 6-17
Using Xmodem Protocol for File Transfer (xmodem) 6-20
  Xmodem Shell Command 6-20
  Sending a File 6-21
  Receiving a File 6-22
Using Pacing Protocol for File Transfer 6-23
  Character Pacing 6-23
  Interval Pacing 6-23

About This Chapter

This chapter describes how to customize and configure the Asynchronous Terminal Emulation (ATE) program to fit your particular needs. Topics covered include modifying local settings, altering connection settings, remapping control keys, and changing the default file. This chapter also discusses two file transfer protocols: xmodem and pacing.

The chapter is written for system users who have data processing experience.

Before You Begin

Unless a connection has been made to a remote terminal, you must start the ATE program. Refer to "Starting ATE" on page 5-14.

The commands in this chapter can be given from either the Unconnected or the Connected Main Menu.

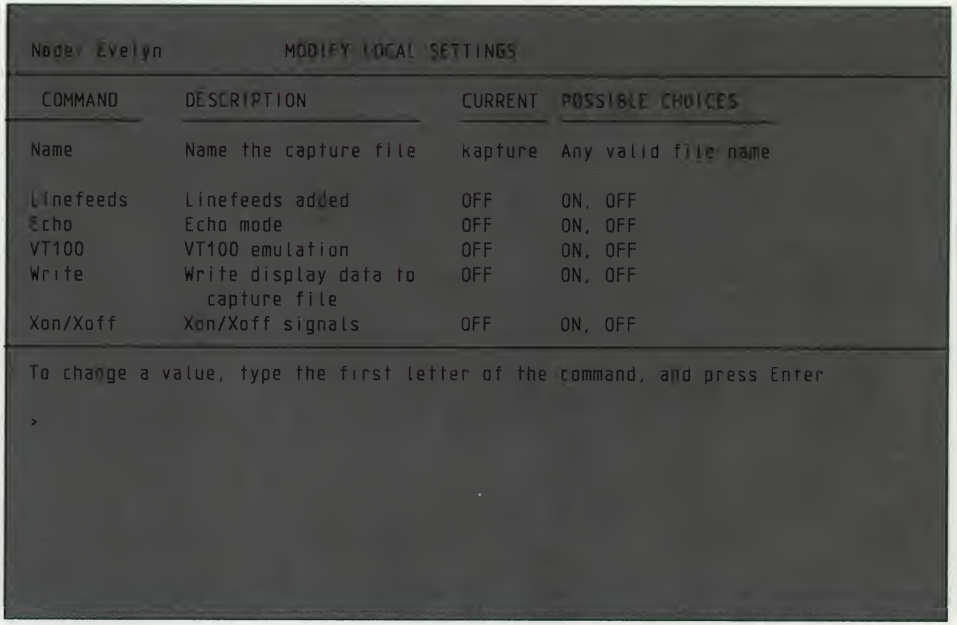
Modifying Local Settings (modify)

The **modify** command lets you change several features of your local system. You can:

- Change the name of the capture file that receives incoming data.
- Switch or toggle the following features *on* or *off*.
  - Add a linefeed character at the end of each line of incoming data.
  - Set echo mode.
  - Emulate a DEC VT100 terminal at the console.
  - Write incoming data to a capture file as well as to the display.
  - Use an Xon/Xoff (transmitter on/off) signal.

If you want to see the Modify Menu to view the features that can be changed and see the current default settings, type the letter **m** after the command prompt on either main menu.

The Modify Menu with the default values that are in effect each time you start ATE looks like this:



The features that can be changed are in the first or Command column. For a description of each feature, see "Description of Features" on page 6-7. Remember that:

- You can change as many features as you want at the same time.
- If you change the **name** variable, you also must supply a value, the new name.
- All other variables are switches that you can turn `on` or `off` by typing the command.

  When you type the command, you switch or toggle the value.

You can invoke the **modify** command from either the Unconnected Main Menu or the Connected Main Menu. If you follow the steps in the instruction box below, you can bypass the Modify Menu.

Modifying Local Settings from a Main Menu

1. On the Main Menu, type **m** *commandinitial* [*value*] (*commandinitial*) after the command prompt.

   For example, to toggle the settings of both the **linefeed** and **echo** features, type:

   ```
   >m l e
   ```

   To change the **name** variable to `schedule` in addition to toggling the **linefeed** and **echo** settings, type:

   ```
   >m n schedule l e
   ```

2. Press the **Enter** key.

   Any data you save now is put into the `schedule` file, and the **linefeed** and **echo** commands are switched to the alternate setting.

Additional Information

- The **( )** (parentheses) indicate that you may repeat the contents as many times as you want.
- The [ ] (brackets) indicate that a value is optional. A value applies only to the **name** variable, since all the other variables are `on/off` switches.
- If you give the **modify** command without typing a variable initial, the Modify Menu displays. You then must type the initial of each variable you want to change after the command prompt.
- If you use a value with any feature other than **name**, an error message appears:

```
062-003  The 'command-name' command is not valid.
         Enter the first letter of a command
         from the list on the menu.
```

If you see this message, either you have typed an incorrect letter or included an invalid value.

Description of Features

name File for incoming data when the **write** command is on, or when the **Ctrl-B** control key is used during a connection.

Options: Any valid file name that is 40 characters or less. The first 18 characters are displayed in the Modify Menu.

Default: *kapture*.

linefeeds Adds a linefeed character after every carriage return in the incoming data stream.

Options: On, Off.

Default: Off

echo Displays your typed input.

With a remote computer that supports echoing, each character you send returns and appears on your display. When **echo** is on, each character displays twice, as you type it and when it returns. When **echo** is off, each character displays only once, when returned from the remote computer.

Options: On, Off.

Default: Off.

VT100 The local console emulates a DEC VT100 terminal so you can use DEC VT100 codes with the remote system. With **VT100** `off`, the local console functions like an IBM RT PC.

Options: On, Off.

Default: Off.

Note: Use this command only if you have the keyboard that came with your system, since some keys are remapped. Any other keyboard gives you unpredictable results. Some DEC VT100 codes, like 132 columns, double-height and -width lines, origin mode, and scrolling regions, are not supported.

write Routes incoming data to the file specified in the **name** command, as well as to the display. This functions like the **Ctrl-B** control key during a connection. Carriage return/linefeed combinations are converted to linefeeds before being written to the capture file. In an existing file, data is appended to the bottom.

Options: On, Off.

Default: Off

Xon/Xoff Controls data transmission at a port as follows:

When an Xoff is received, transmission stops.

When an Xon is received, transmission resumes.

An Xoff is sent when the receive buffer is nearly full.

An Xon is sent when the buffer is no longer full.

Options: On, Off.

Default: Off

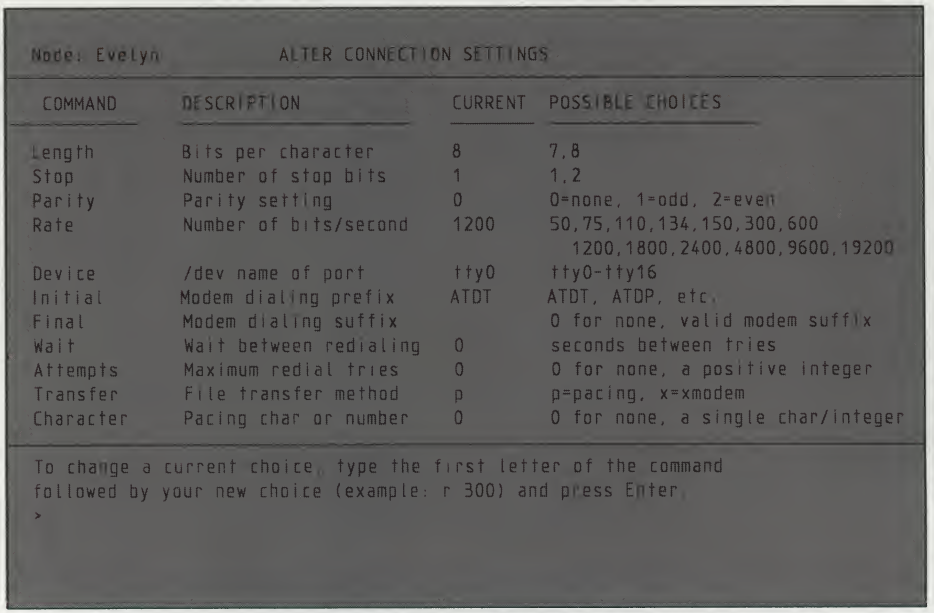
Altering Connection Settings (alter)

The **alter** command lets you change several data transmission characteristics, including:

- Bit length and rate
- Stop and parity bits
- Port name
- Modem dialing prefixes and suffixes
- Waiting time and retry limits
- File transfer method
- Pacing character or delay time.

If you want to see the Alter Menu to view the characteristics that can be changed and the current default settings, type the letter **a** after the command prompt on either main menu.

The Alter Menu with the original default values in effect each time you start ATE looks like this:



The characteristics that can be changed are in the first or Command column. For a description of each characteristic, see "Description of Characteristics" on page 6-12. Remember that:

- You can change as many features as you want at a time.
- You must type a new value for each characteristic you change. For example, if you want to change the number of bits per second (the rate) to 9600, type:

  ```
  >r 9600
  ```

You can invoke the **alter** command from either the Unconnected Main Menu or the Connected Main Menu. If you follow the steps in the instruction box below, you can bypass the Alter Menu.

Altering Connection Settings from a Main Menu

1. On the Main Menu, type **a** *commandinitial value* (*commandinitial value*) after the command prompt. For example, to change the **rate**, **wait**, and **attempt** values, type:

   ```
   >a r 9600 w 5 a 1
   ```

2. Press the **Enter** key.

Additional Information

- The **( )** (parentheses) indicate that you may add as many variables as you want.
- Type the first letter of the variable that you want to change, followed by the value you select.
- Press **Enter** again to leave the Alter Menu.

Description of Characteristics

length Number of bits in a data character. The length must match the length expected by the remote system.

Options: 7 or 8.

Default: 8

stop Number of stop bits appended to a character to signal the end of that character during data transmission. This number must match the number of stop bits used by the remote system.

Options: 1 or 2.

Default: 1

parity Checks whether a character was successfully transmitted to or from a remote system. Must match the parity of the remote system.

For example, if you select even parity, when the number of 1 bits in the character is odd, the parity bit is turned on to make an even number of 1 bits.

Options: 0 = none, 1 = odd, 2 = even.

Default: 0

rate Bit rate (bits per second). Determines the number of bits transmitted per second. The speed must match the speed of your modem and that of the remote system.

Options: 50, 75, 110, 134, 150, 300, 600, 1200, 1800, 2400, 4800, 9600, or 19200.

Default: 1200

device Name of the asynchronous port used to make a connection to a remote system.

Options: tty0 - tty16, or locally created port names. Only the first 8 characters appear in the Alter Menu.

Default: tty0

initial Dial command that must precede the telephone number when you autodial with a modem. Consult your modem user's guide for the proper dial commands.

Options: ATDT, ATDP, etc. Only the first 8 characters appear in the Alter Menu.

Default: ATDT

final Dial command that must follow the telephone number when you autodial with a modem. Consult your modem user's guide for the proper dial command.

Options: 0 = none, valid modem suffix. Only the first 8 characters appear in the Alter Menu.

Default: None

wait Tells the **redial** command in the ATE program to wait *n* seconds between redialing attempts. The wait period does not begin until the connection attempt times out or until you interrupt it. If **attempts** is set to 0, no redial attempt occurs.

Options: 0 = none, a positive integer.

Default: 0

attempts Maximum number of times ATE tries to redial to make a connection. If **attempts** is set to 0, no redial attempt occurs.

Options: 0 = none, a positive integer.

Default: 0

transfer Type of asynchronous protocol that transfers files during a connection.

Pacing
File transfer protocol that controls the data transmission rate by waiting for a specified character or a certain number of seconds between line transmissions. Helps prevent loss of data when the transmission blocks are too large or are sent too quickly for the system to process.

For more information, see "Using Pacing Protocol for File Transfer" on page 6-23.

Xmodem
An 8- bit file transfer protocol that detects data transmission errors and retransmits the data. For more information, see "Using Xmodem Protocol for File Transfer (xmodem)" on page 6-20.

Options: p = pacing, x = xmodem

Default: p

character
Specifies the type of pacing to be used.

Character Signal to transmit a line. (Select one character).

Integer Number of seconds the system waits between each line it transmits. (Select one integer). The default value is 0, indicating a pacing delay of 0 seconds.

Default: 0

Changing (Remapping) the Control Keys

The control keys were introduced in "Functions of Control Keys" on page 5-9. The mapping for these key combinations, set in the default file **ate.def**, follows:

capture key
Starts or stops capturing (saving) the data displayed on your screen during an active connection. This key has a switching or toggle effect.

Options: Any ASCII control character

Default: ASCII octal 002 (STX). This is the **Ctrl-B** key combination on the RT PC keyboard.

main-menu key
Returns the Connected Main Menu so you can issue a command during an active connection. This control key functions only from the connected state.

Options: Any ASCII control character

Default: ASCII octal 026 (SYN). This is the **Ctrl-V** key combination on the RT PC keyboard.

previous key
Displays the previous screen any time during the program. The screen that displays varies, depending on the screen in use when you press the previous-screen key. See "Functions of Control Keys" on page 5-9 for more information.

Options: Any ASCII control character

Default: ASCII octal 022 (DC2). The ASCII control character is mapped to the AIX interrupt signal. This is the **Ctrl-R** key combination on the RT PC keyboard.

Changing or remapping the control keys may be necessary if you have two applications in which control keys conflict. For example,

if the control keys mapped for the ATE program conflict with those in your text editor, you may want to remap the control keys for ATE.

A system user with some programming experience can remap the control keys by editing the default file **ate.def**. See "Changing Your Default File" on page 6-17.

ASCII Control Characters

The ASCII control character you select may be in octal, decimal, or hexadecimal format.

octal	000 through 037
	The leading zero is required.
decimal	0 through 31
hexadecimal	0x00 through 0x1F
	The leading 0x is required. The x may be uppercase or lowercase.

Changing Your Default File

The first time you run the ATE program, it creates a file named **ate.def** in the current directory. This file contains default values for:

Data transmission characteristics
See "Description of Characteristics" on page 6-12 for descriptions.

Local system features
See "Description of Features" on page 6-7 for descriptions.

Dialing directory file
See "Creating a Dialing Directory File" on page 5-23 for description.

Control keys See "Changing (Remapping) the Control Keys" on page 6-15 for descriptions.

The default file that is created the first time ATE runs looks like this:

```
LENGTH          8
STOP            1
PARITY          0
RATE            1200
DEVICE          tty0
INITIAL         ATDT
FINAL
WAIT            0
ATTEMPTS        0
TRANSFER        p
CHARACTER       0
NAME            kapture
LINEFEEDS       0
ECHO            0
VT100           0
WRITE           0
XON/XOFF        0
DIRECTORY       /usr/lib/dir
CAPTURE_KEY     002
MAINMENU_KEY    026
PREVIOUS_KEY    022
```

A system user with some data processing experience can edit the **ate.def** file with any AIX text editor to change the values of these characteristics.

How to Edit the Default File

1. Leave ATE.
2. Enter the text editor and display the **ate.def** file.
3. Delete all variables which you *do not* want to change.

   This simplifies reading the file, since the system ignores variables that remain the same as the system's defaults.
4. Change the values for those variables you want to change.
5. Save the file.

We now give an example. Assume that you want to change:

RATE to 300 bps.
DEVICE to `tty3`
DIRECTORY to `my.dir`
TRANSFER to `x` (xmodem)

The **ate.def** file that you create looks like the following:

```
RATE            300
DEVICE          tty3
TRANSFER        x
DIRECTORY       my.dir
```

From now on when you start using ATE, the program looks for a file in the current directory named **ate.def**. Then it reads the values in the file and changes the system's defaults to those you set.

Additional Information

- Variable names must be in uppercase and spelled exactly as they appear in the original default file.
- Type only one variable per line.
- If you enter an incorrect value, you receive a system message but the program continues, using the default value.

Using Xmodem Protocol for File Transfer (xmodem)

Xmodem protocol is an 8-bit transfer protocol that can detect data transmission errors and then retransmit the data. The work station that sends data must wait until the remote system sends a signal that it is ready to receive data.

When the receiver gets data, it returns an acknowledgement to the sender. The ATE receiver times out if it does not receive data 90 seconds after file transfer is initiated.

Xmodem Shell Command

A shell command **xmodem**, shipped with ATE, starts an xmodem file transfer protocol on a remote RT PC. This command is used in combination with the **send** or **receive** command on the Connected Main Menu.

Sending and receiving with **xmodem** are complementary operations. One system must be set to send while the other is set to receive.

You can interrupt the **xmodem** shell command by pressing the **Ctrl-X** key combination.

Format

`xmodem` *-opt filename*

xmodem Transfers files between computer systems that have ATE installed.

-opt The option you use with the **xmodem** command:

- **-s** Send data to the local RT PC.
- **-r** Receive data from the local RT PC.

filename The name of the file you send or receive.

Sending a File

After a connection to a remote RT PC is established, you can send a file from the local system to the remote RT PC with the following command:

Sending a File with Xmodem Protocol

1. Type `xmodem -r` *filename.*

   For example, to send `mayfile` to a remote RT PC, type:

   ```
   xmodem -r mayfile
   ```

2. Press **Ctrl-V**.

   The Connected Main Menu displays.

3. Select the **send** command.

   ```
   -s mayfile
   ```

After the file is transferred, the connection screen displays.

Receiving a File

After a connection to a remote RT PC is established, you can receive a file from the remote RT PC with the following procedure:

Receiving a File with Xmodem Protocol

1. Type `xmodem -s` *filename.*

   For example, to receive the `infile` from the remote RT PC, type:

   ```
   xmodem -s infile
   ```

2. Press **Ctrl-V.**

   The Connected Main Menu displays.

3. Select the **receive** command.

   ```
   -r infile
   ```

After the file is transferred, the connection screen displays.

Using Pacing Protocol for File Transfer

Pacing is a file transfer protocol required by some systems. It controls data transmission by waiting for a specified character or waiting a specified number of seconds between lines. This protocol prevents the loss of data when the block size is too large or data is sent too quickly for the system to process.

Use this protocol to send text files only. Files containing embedded control characters cannot be transferred.

Asynchronous Terminal Emulation supports two types of pacing control protocols: character pacing and interval pacing.

Character Pacing

The **send** protocol transmits data from the file until it finds a linefeed. It then sends out a carriage return and waits to receive the pacing character before sending more data. It times out if it does not receive the correct pacing character in 30 seconds.

The **receive** routine sends a pacing character as soon as it is started. It times out if it does not receive data within 30 seconds.

When data transmission starts, the **receive** routine sends a pacing character to the sender whenever it finds a carriage return in the data. The program ends when it receives nothing for 30 seconds.

Interval Pacing

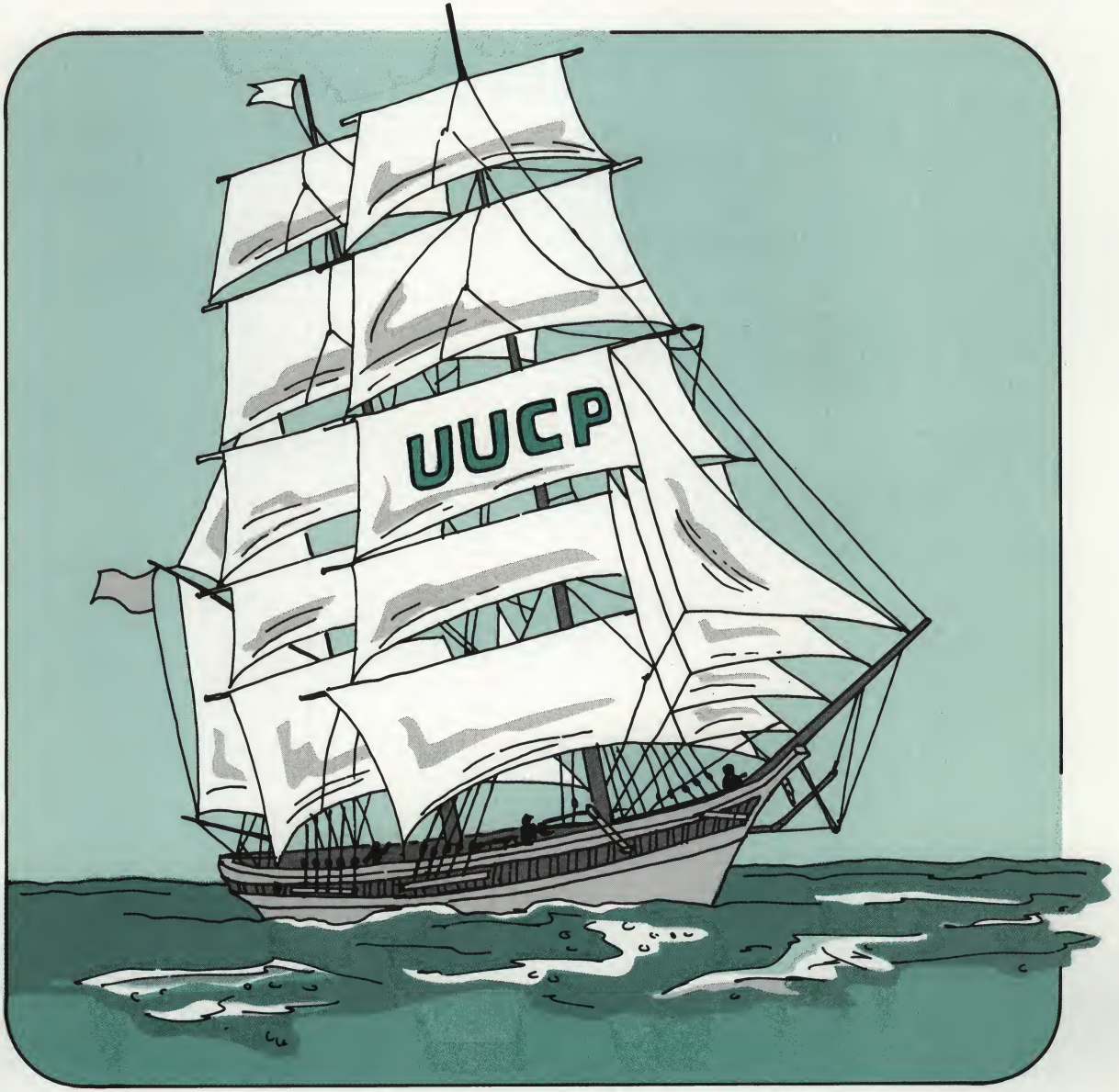
The **send** routine begins to transfer data as soon as it is started. When it finds a linefeed, it converts it to a carriage return and waits a specified time before it sends the next line.

The **receive** routine looks for data as soon as it is started, and times out if it does not receive data within 30 seconds.

Additional Information

UNIX files traditionally contain only linefeeds (called ***newlines***) to delimit the end of a line. DOS or CP/M files usually use only carriage returns. For this reason, the pacing **send** protocol converts linefeeds to carriage returns. The pacing **receive** protocol converts any linefeed/carriage return combinations or single carriage returns to linefeeds before saving the file.

Chapter 7. Using the UNIX-to-UNIX Copy Program (UUCP)



CONTENTS

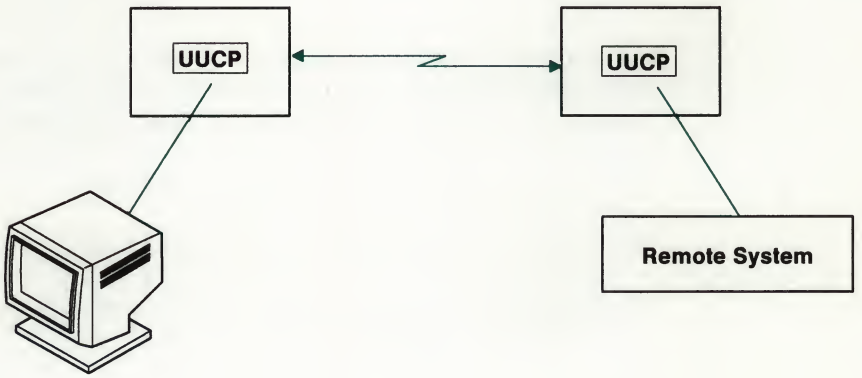
About This Chapter 7-3
Introduction to UUCP 7-4
Identifying Compatible Systems (uuname) 7-5
Sending and Receiving Files (uucp) 7-7
  Path Names 7-9
  Local Transfers 7-10
  Remote Transfers 7-12
Controlling File Access at the Local System 7-14
  Sending Files to a Specific ID (uuto) 7-15
  Locating Files for a Specific ID (uupick) 7-17
Running Remote Commands (uux) 7-20
Getting Status Information (uulog) 7-23
Removing Old Files (uuclean) 7-24
Establishing a Subnetwork (uusub) 7-26
Running Commands Automatically (uudemon) 7-28
  uudemon.hr File 7-29
  uudemon.day File 7-30
  uudemon.wk File 7-31
How UUCP Works 7-32
  Files and Directories 7-32
  Sending and Receiving Files 7-34
How UUX Works 7-37

About This Chapter

This chapter tells how you can use the UNIX-to-UNIX Copy Program (UUCP), a component of the Extended Services facility of AIX, to:

- Send and receive files locally or transmit files between the local system and a remote system.
- Run remote commands.
- Manage network activities.

The following illustration provides a general overview of the communication.



Since UUCP is a complex AIX communication facility, you should be familiar with UNIX or AIX and with the IBM RT PC before you use UUCP.

To install UUCP on your system, select UUCP (item 6) from the list of choices on the install menu (diskette 1 of Extended Services). Then, follow the instructions that appear on the display.

See Appendix A for an overview of the installation procedure. If you need additional information, see *Installing and Customizing the AIX Operating System.*

Introduction to UUCP

The UNIX-to-UNIX Copy Program (UUCP) is a group of programs and files within the Extended Services facility of AIX that functions as an AIX ***background process***. This means that once a UUCP task is running, you can use your work station for other jobs. For example, you can send a file to a remote system for printing and, while it prints, edit another document at your work station.

In addition to communication within a local system, with UUCP two UNIX-based computer systems can communicate directly, over a hardwired line, or indirectly, over a telephone line.

UUCP has two communication features:

Call-out Sends files and commands to another system

Call-in Receives files and commands sent from another system.

You can have both types of communication on your system if you have a line dedicated to UUCP.

Identifying Compatible Systems (uuname)

Each remote system with which you communicate must have a UNIX-based operating system, like AIX, and be connected to your local system. These remote systems are identified in the **L.sys** file by the person who customizes your system. See "System Names (L.sys)" on page 8-12 for more information.

To identify the names of systems with which you can communicate, use the **uuname** command.

Format

`uuname` [*option*]

uuname
Displays the names of systems with which you can communicate. One flag, called an option in the command format, can be added.

-l Displays the name of your local system.

How to Identify Compatible Systems

1. Type the **uuname** command after the **$** prompt on the command line.

   ```
   uuname
   ```

2. Press the **Enter** key.

   The names of other systems appear on your display:

   ```
   sitename
   alec
   bravo
   ```

If you want to check the name of your local system:

1. Add the **-l** flag to the **uuname** command:

   ```
   uuname -l
   ```

2. Press the **Enter** key.

   The name of your system appears on the display:

   ```
   tempest
   ```

Sending and Receiving Files (uucp)

The basic task of the UUCP facility is to send and receive files. With the **uucp** command and its options you can send and receive files:

- Within your local system
- Between a remote system and the local one
- Between two remote systems.

You can also use an option to request that a remote computer run the **uucp** command. This request is granted only if the remote system is configured to run that command remotely. For more complete discussion, see the **-e** option below, "Running Remote Commands (uux)" on page 7-20, and "Remote Commands (QTCMDS)" on page 8-20.

Format

```
uucp [options] sourcefile(s) destinationname
```

uucp Transfers files between or within UNIX-based systems. Flags, called options in the command format, can be added.

-C With this flag, the source file is temporarily held in the spool directory while the **uucico** program contacts the remote computer to complete the file transfer.

-c Stops the creation of an intermediate data file in the spool directory. With this flag, the source file goes directly to the destination file. This is the default. You need this flag if you used the **-C** flag and want to change back to the default situation.

-d Creates the directories needed to copy a file. These directories are provided with the UUCP facility. If you need to recreate a directory, use the **-d** flag. For additional information, see "How UUCP Works" on page 7-32.

-e*systemname*
Requests that a remote system run the **uucp** command. If the remote system is configured to run **uucp**, the **uux** program creates and sends the request. See "Running Remote Commands (uux)" on page 7-20 and "Remote Commands (QTCMDS)" on page 8-20 for more information.

-m[*filename***]**
Sends a message when a remote transfer is complete. The message goes to your mail box, **/usr/mail/***username* or to the file you specify. No message is sent for a local transfer.

-n*username*
Notifies recipient *username* on a remote system that a file has been sent. No message is sent for a local transfer.

sourcefile
Location of the file you want to send or receive. See "Path Names" on page 7-9.

destinationname
File or directory where the copy goes. See "Path Names" on page 7-9.

For information on other **uucp** options, see *AIX Operating System Commands Reference.*

The sections that follow discuss path names and the procedures for local and remote file transfers.

Path Names

A path name may be one of the following:

Full *pathname*
: Lists all the directories along the path from the root to the specific file, ending with the file name. By convention, the names are separated by slashes.

Directoryname
: **uucp** adds any needed directory names between the current directory and the file, and creates a file name from the last part of the source file.

Partial *pathname*
: Lists directories between the current working directory and the file name. If you are already in the `/usr/bin/tony/mice` directory, then keying in `diets` (with no leading slash) identifies the file `/usr/bin/tony/mice/diets`.

~user pathname
: The *~user* means the login directory of the user.

~/ filename
: Files in the public directory. Usually **/usr/spool/uucppublic** on the remote system--the public directory defined by UUCP for sending and receiving information.

Systemname! *pathname*
: Path to a file on another system. An example follows:
`sysabc!/research/new`

(*Systemname*!) *pathname*
: Path to a file on another system that goes through one or more other systems. An example follows:
`sysx!sysy!sysz!/research/cells`

Local Transfers

In a local transfer, the **uucp** command functions like **cp**, the AIX **copy** command. You can copy files in the same or in different directories within the system. (See *AIX Operating System Commands Reference* for more on **cp**).

You need to know the path names of the source file, or original location, and the destination name, the file or directory where the file copy goes.

Using uucp for Local Transfers

1. Type **uucp** *sourcefile(s) destinationname.*

   The following example copies `prog.b` to `prog.c` in the same directory. If `prog.c` does not exist, the system creates it.

   ```
   uucp prog.b prog.c
   ```

   This next example copies the `jones` file into the `clients` directory. The system creates a file called `jones` in the new directory, so the new path is `clients/jones`.

   ```
   uucp jones clients
   ```

   The last example shows several files being copied to the `/u/tom` directory:

   ```
   uucp listing clients/smith /u/tom
   ```

2. Press the **Enter** key.

Additional Information

- The source file name can contain special shell characters (?, *, [, and]) which the remote shell interprets. To keep the local shell from interpreting the special characters before they are sent, surround the line with " (double quotes) or \\ (backslashes).

  ```
  "/usr/bin/tony/?"
  ```

- Do not use special shell characters in the destination name.

- The **uucp** command preserves any execute permissions it finds in the directories it searches.

- The permissions granted by **uucp** give read and write permission to the owner, the group, and all others. This can be expressed as 0666 permission. To change the permissions, refer to the **chmod** command in *AIX Operating System Commands Reference* and to the discussion on permissions in *Using and Managing the AIX Operating System.*

- When **uucp** is run remotely, it displays a job number for each remote use.

Remote Transfers

To make remote transfers, add the remote *systemname*! (exclamation point) to the path of the file that receives the copy.

Remote transfers usually are restricted to public spool directories.

Sending Files

You can send files directly to a remote system or through one or more intermediate systems:

Sending Files to a Remote System

1. Type **uucp** [*options*] *sourcefile(s)* *systemname*!*destinationname*

   For example, the following sends a copy of your `/meteors/small.c` file to the `/solar/stats.c` file in the public directory on the remote system `galaxy`.

   ```
   uucp /meteors/small.c galaxy!~/solar/stats.c
   ```

   The public spool directory **/usr/spool/uucppublic** is indicated by the ~ in the destination path.

2. Press the **Enter** key.

To send files through intermediate systems, add each remote *systemname*! (exclamation point). In the following example, there is one intermediate system, `milkyway!`:

```
uucp /meteors/small.c milkyway!galaxy!~/solar/stats.c
```

Receiving Files

You can receive files directly from a remote system or through one or more intermediate systems, if **uucp** has write permission in the directory that receives the files.

Receiving Files from a Remote System

1. Type **uucp** [*options*] *systemname*!*sourcefile(s) destinationname*

   To request the `/cells/type1` file from system `biochem!` and store it in your `!/drjjm/research` file on the local system, type:

   ```
   uucp biochem!/cells/type1 !/drjjm/research
   ```

   Your local system is identified by an **!** (exclamation point), but does not need a system name.

2. Press the **Enter** key.

Controlling File Access at the Local System

The **uuto** and the **uupick** commands give the local system a tool for controlling access to its files. With these commands, files transferred from the local system must be sent to a specific user ID on the other system. An overview of the method follows:

1. The sender uses the **uuto** command to send files to a specific user ID.
2. The **uuto** program sends the files to **/usr/spool/uucpublic** (**PUBDIR**), the UUCP public directory, and notifies the recipient by a mail message.
3. The recipient gives the **uupick** command.
4. The **uupick** program searches **PUBDIR** for files sent to the recipient and notifies the recipient about each file it locates.
5. Through a series of options, the recipient saves or deletes each file.

The **uuto** and **uupick** commands may also be used to transfer files to a specific ID within the local system.

More information on **uuto** and **uupick** follows. Also refer to *AIX Operating System Commands Reference.*

Sending Files to a Specific ID (uuto)

The **uuto** command uses the following format.

Format

`uuto` [*options*] *filename destinationname*

uuto Sends files to a specific user ID. The following flags, called options in the format, can be added:

-m Notifies the sender by mail when a remote transfer is complete.

-p Sends the source file to the spool directory for transfer to **/usr/spool/uucppublic** (**PUBDIR**). Without the flag, the source file is sent directly to **PUBDIR**. In either case, when the **uupick** command is given, the **uupick** program completes the transfer to the specified user ID.

filename
File sent to a specific ID.

destinationname
Location to which the files are sent. For a remote system, the destination name must be `systemname!username`. For a local system, the destination name is the username.

Sending Files to a Specific User ID

1. Type **uuto** [*options*] *filename destinationname* after the **$** prompt on the command line.

   For example, to send file `/usr/bin/data/private` to `smythe` on remote system `bravo`, type the following:

   ```
   uuto /usr/bin/data/private bravo!smythe
   ```

2. Press the **Enter** key.

   The **uuto** command sends `smythe` a mail message when the file arrives at system `bravo`. Smythe then issues the **uupick** command to complete the transfer. For information on **uupick**, see "Locating Files for a Specific ID (uupick)" on page 7-17.

Additional Information

- The **uuto** command is useful if a remote-system user is eligible to receive a file, but finds access restricted to local users. The remote user can request that the local system send the file to the remote user's ID.
- If **uuto** is used for a transfer within the local system, omit the system name in the command format. No mail message is sent in a local transfer.

Locating Files for a Specific ID (uupick)

Files sent to your ID by the **uuto** command go to **/usr/spool/uucppublic** (PUBDIR) and wait to be transferred to your system.

When you receive a mail message that the files are in PUBDIR, use the **uupick** command to receive and handle the files.

Format

```
uupick
```

Receiving and Handling Your Files

How to Receive Your Files

1. When you receive a mail message that a file is waiting in **PUBDIR**, type **uupick** after the **$** prompt on the command line.

   ```
   uupick
   ```

2. Press the **Enter** key.

   When you receive the file, a message appears in this format: `from system` *systemname*: [`file` *filename*] [`dir` *directoryname*], and is followed by a **?** prompt.

   For example, if `smythe` sent you `/info/local`, you receive the following message:

   ```
   from system bravo: file /smythe/info/local
   ?
   ```

3. After the **?** prompt, enter a command from the following list to indicate how you want the file handled.

   Continue until all the files have been handled or use **q** or **EOF** to stop reviewing files.

To display the list of choices for handling your files, type an ***** (asterisk).

Options for Handling Your Files

Carriage return	Go to the next file.
a[*dir*]	Move all files to directory *dir*. If the *dir* field is blank, the files move to the current directory.

d	Delete this file.
m[*dir***]**	Move the file to directory *dir*. If the *dir* field is blank, the file moves to the current directory.
p	Display contents of file on screen.
q (or EOF)	Lets you quit without handling any files.
!*command*	Escape to the shell to run *command* and then return to **uupick**.

Running Remote Commands (uux)

The **uux** command can run a command on a remote system, if the remote system is configured to run the command you select.

Each Unix-based system has a **/usr/lib/uucp/QTCMDS** file that lists the commands you can access remotely. If the command you want to run is listed in the file, the **uux** program creates and sends the request. See "Remote Commands (QTCMDS)" on page 8-20.

A remote system may restrict certain commands to local use. Some systems only permit access to the **mail** command.

Format

`uux [options] commandstring [> \(destinationname\)]`

uux Runs a command on another AIX-based system. One or more flags, called options in the format, may be added.

-n User is not notified by mail, whether the command completes successfully or fails. Do not use with **-z**.

-z User is not notified by mail if the remote command completes successfully. Do not use with the **-n** flag.

commandstring
One or more commands with *filenames*, as needed. The first command may be prefixed by *systemname***!** to indicate the system on which the command is requested. The default is the local system.

> \(*destinationname***\)**
Optional parameter, indicating the system and file to store the output of the **uux** program. The default is the local system.

Refer to *AIX Operating System Commands Reference* for additional options and more information on the **uux** command.

How to Run a Remote Command

1. Type **uux** *command string* [> \(*destinationname*\)]

   To send the `/book/chapter01` file to the `bravo` system for printing, type the following:

   ```
   uux bravo!print /book/chapter01
   ```

   For this request to be successful, the **print** command must be listed in the **/usr/lib/uucp/QTCMDS** file. This example does not include a destination name since the output is to be stored on the system that runs the command.

2. Press the **Enter** key.

Additional Information

- If you request a command that the remote system cannot run, you receive a mail message from the remote system.
- Place the system name before the first command in an entry. Other commands that request the same system follow in the same entry.

  For example, to have the `bravo` system run both the **mail** and the **print** commands, place both commands in the same entry:

  ```
  uux bravo!mail/print /book/chapter01
  ```

- To run commands on more than one system, type the information for each system on a separate line.

  ```
  uux bravo!print /book/chapter01
  uux satellite!print /book/chapter02
  ```

- To run a command on the local system using files from two different remote systems, substitute ! for *system* ! for the local system `abc`, and add *system* ! before each */filename.*

For example, to run the **diff** command on the local system to generate a list of differences in files `/pat/sun` and `/jon/sun` from systems `xox` and `yoy`, type the following:

```
uux "!diff xox!/pat/sun yoy!/jon/sun"
```

- To store the above list in file `/sun.diff` on the local system, add a destination name. The complete command is:

  ```
  uux "!diff xox!/pat/sun yoy!/jon/sun > !/sun.diff"
  ```

- To store the output file in system `zoz`, type `abc!` for the local system and add the new destination name:

  ```
  uux abc!diff xox!/pat/sun yoy!/jon/sun \(zoz!/sun.diff\)
  ```

  The sequence `\( ... \ )` must surround output files as well as parts of a command that contain an exclamation point.

- Names for source file(s) and the destination name may be a full path name, a partial path name, or a path name preceded by *~user*. Replace *user* with a login name that refers to the user's login directory.

- A source path name may contain shell special characters (?, *, [, and]) that the remote system can interpret. Enclose special characters in \\ or " so the local shell cannot interpret the characters before the command is sent to the remote system.

  Do not use special characters in destination names.

- If you use other shell characters (<, >, ;, and |), either put \\ or " around the individual character or the entire command string. Do not use shell redirection characters (< < and > >) because they do not work in the UUCP facility.

For system administrators who need to know how the **uux** program works, see "How UUX Works" on page 7-37.

Getting Status Information (uulog)

The **uulog** command appends individual LOG files to the main log file, **/usr/spool/uucp/LOGFILE**.

Status information about each transaction should go directly into the **/usr/spool/uucp/LOGFILE** file each time UUCP is used. When more than one UUCP process is running, LOGFILE cannot be accessed, so status information goes into a file with a LOG prefix that covers just the one transaction. The **uulog** command put all the LOG files together.

You can access **uulog** directly or append the files automatically by including instructions in the **/usr/lib/crontab** files. The **cron** daemon gives the command at the time interval you select. For information about **crontab** and **cron**, consult *AIX Operating System Commands Reference*. A sample file is included in "Running Commands Automatically (uudemon)" on page 7-28.

You can also display a summary of **uucp** and **uux** transactions by user or system with the **uulog** command, since all these transactions are logged in **/usr/spool/uucp/LOGFILE**.

Format

```
uulog [options]
```

uulog Appends individual LOG files to LOGFILE.

-u*user* Displays a summary of **uucp** and **uux** transactions in LOGFILE for the user named in the flag.

-s*system* Displays a summary of **uucp** and **uux** transactions in LOGFILE for the system named in the flag.

Removing Old Files (uuclean)

The **uuclean** command deletes files in the spool directory that are more than a predefined age. It also notifies the user who requested the work that the files have been removed.

The default value is 72 hours, since files left in the spool directory for three days are usually for work that cannot be completed.

Deleting files is automatic if you include instructions in the **/usr/lib/crontab** files and submit the schedule to the **cron** daemon. Sample files are included in "Running Commands Automatically (uudemon)" on page 7-28. For information about **crontab** and **cron**, consult *AIX Operating System Commands Reference.*

Format

```
uuclean [options]
```

uuclean Deletes files in the spool directory that are more than the default age (72 hours) or the age set with the **-n** flag. All files that age are deleted unless you identify the prefixes of files to delete with the **-p** flag.

-n*hours* Deletes files older than *hours.*

-p*prefix* Limits file deletion to those age-eligible files identified by *prefix*. The prefix is that part of a file name that follows a **.** (dot). You can include this flag up to ten times, to set ten prefixes for deletion.

Additional Information

You have direct access to the **uuclean** command, but it is common practice to have the **cron** daemon automatically call the command. This way, you cannot forget to clean out old files.

Usually the **uudemon.day** daemon is set to call the **uuclean** command. See "Running Commands Automatically (uudemon)" on page 7-28 for an example of a file that includes **uuclean**.

Establishing a Subnetwork (uusub)

The **uusub** command helps manage network activities by establishing and maintaining subnetworks, and collecting statistics on network connections and traffic.

The collection of statistics is automatic if you include instructions in **crontab** files and submit the schedule to the **cron** daemon. A sample file is included in "Running Commands Automatically (uudemon)" on page 7-28. For information about **crontab** and **cron**, consult *AIX Operating System Commands Reference.*

Format

```
uusub [options]
```

uusub Defines subnetworks, adding and deleting systems as needed. Collects and displays network connection and traffic statistics. Several flags, identified as options in the format, can be added.

-a*system* Adds a system to the subnetwork.

-d*system* Deletes a system from the subnetwork.

-c*system* Gets connection statistics from the **/usr/lib/uucp/L_sub** file.

-c all If **all** is used with the **-c** flag, instead of a system name, **uusub** collects statistics from all the systems in the subnetwork.

-f Clears the connection statistics

-l Gets traffic statistics from the **/usr/lib/uucp/R_sub** file.

-u*hours* Time for collecting traffic statistics.

Additional Information

- The **cron** command usually starts the **uusub** command once a day, using the following:

  ```
  uusub -c all -u 24
  ```

  This example collects connection statistics from all the systems in the subnetwork and traffic statistics for the past 24 hours.

- Connection statistics include:

  - Number of times the local system tried to call a system since the last flush
  - Number of successful connection attempts
  - Latest successful connect time
  - Number of unsuccessful connection attempts

  A connection attempt may be unsuccessful because no device was available, a login failed, or there was no response.

- Traffic statistics include the:

  - Number of files sent and received
  - Number of bytes sent and received in the time indicated in the the latest **uusub-u** command.

For more information on **uusub**, see *AIX Operating System Commands Reference.*

Running Commands Automatically (uudemon)

The three sample files that follow contain typical routines for automatic maintenance of the UUCP facility. The following commands are included in one or more of the examples:

- **uucico**
- **uulog**
- **uuclean**
- **uustat**
- **uusub**

If you place the entries in the **/usr/lib/crontab** file and submit the schedule to the **cron** daemon, the commands are run at appropriate times. It is customary to run **cron** from the system initialization process through the file **/etc/rc**.

For information about **crontab** and **cron**, consult *AIX Operating System Commands Reference.*

uudemon.hr File

The sample file that follows contains instructions for setting the local system to primary mode and obtaining status information every hour.

```
        # 'perform every hour on the 56-minute mark'
        PATH=:/bin:/usr/bin

        cd /usr/lib/uucp
*** The following starts a program that sets the ***
*** local system to primary mode. ***
        uucico -r1

*** The following collects all log information into ***
*** one file: ***
        uulog
```

uudemon.day File

The sample file that follows contains instructions for cleaning up the execute directory, removing old status entries, gathering and logging status information, and removing old files from the public directory on a daily basis:

```
      # 'perform once per day at 0400 hours'
      PATH=:/bin:/usr/bin

      cd /usr/lib/uucp
*** The following lines clean up the execution ***
*** directory: ***
      uuclean -p -m -n168 >/dev/null 2>/dev/null
      uuclean -d.XQTDIR -p -n72 >/dev/null 2>/dev/null

*** The following cleans up old status entries: ***
      uustat -c168 >/dev/null 2>/dev/null

*** The following gathers statistics: ***
      uusub -u24

      cd /usr/spool/uucp
*** The following places LOGFILE in a temporary ***
*** location: ***
      mv LOGFILE temp

*** The following adds any unique entries to the ***
*** weekly log: . ***
      uniq -c temp >> Log-WEEK

*** The following deletes the daily log: ***
      rm temp

      cd /usr/spool/uucppublic
*** The following removes any file in the public ***
*** directory that has not been modified in 30 days: ***
      find -type f -mtime +30 -exec rm -f [] \ ;
```

uudemon.wk File

The following sample file establishes a procedure for removing and saving old logs on a weekly basis:

```
     # 'perform once a week, Sunday, at 0530 hours'
     PATH=:/bin:/usr/bin

     cd /usr/spool/uucp
*** The following removes old packed logs: ***
     rm -f o.Log-WEEK* o.SYSLOG*

*** The following moves last week' log into  ***
*** old logs and adds a date:   ***
     mv Log-WEEK o.Log-WEEK
     (date; echo ============) >> o.Log-WEEK

*** The following moves last week's system log ***
    into old logs: ***
     mv SYSLOG o.SYSLOG

*** The following packs the old logs: ***
     pack o.Log-WEEK o.SYSLOG
```

How UUCP Works

The following information is for system administrators who need to know how the **uucp** command works.

The two primary tasks of UUCP are to:

- Transfer files within and between systems, using **uucp** and related programs
- Request a remote system to run AIX commands for another system, using **uux** and related programs.

File transfer begins with the **uucp** command. This tells the **uucp** program to collect the files you want to transmit. The **uucp** program then gives the files to another program, **uucico**, to make the transfer.

Running AIX commands on a remote system begins with the **uux** command. This tells the **uux** program to create and collect the files needed to run commands remotely. Some of these files contain instructions for running the commands. The **uuxqt** program carries out the instructions.

Files and Directories

UUCP uses three types of files to transfer data between systems:

Data files Data to send to remote systems.

Work or command files
Directions for the transmitting program, **uucico**.

Execute files Instructions for running commands that require the resources of a remote system.

Four system directories are needed to transfer files and to run commands remotely:

/usr/lib/uucp Contains system files and some executable programs.

/usr/spool/uucp
Contain files needed to run the **uucp** command and store temporary or spool files.

/usr/spool/uucppublic
Temporarily stores files in transit and files created for processing. These files are held until the destination directory indicates that it can receive them.

/usr/spool/uucp/.XQTDIR
Contains execute files with lists of commands remote systems may run.

Sending and Receiving Files

The **uucp** program classifies each **uucp** request as local or remote. It then adds a line to an existing work file for each remote request, or creates a new file if the current file is full.

Sending Files to Remote Systems

To send files to a remote system, a work file entry must have the following fields, with a blank between each field:

- S (for send)
- Full path name of the source file
- Full path name of the destination name or ~ *user pathname*
- User's login name
- A - (hyphen) followed by a **-d**, an **-m**, or an **-n**.
- Name of the data file in the spool directory.

  A copy of each source file goes into a *data file* in the spool directory, unless you specified the **-c** flag. Use a dummy name, **D.0**, with the **-c** option.
- File mode bits of the source file in octal print format, such as 0666.
- User to notify when the transfer finishes, if you used the **-n** option.

Receiving Files from Remote Systems

To receive files from a remote system, a work file entry must have the following five fields, with a blank between each field:

- R (for receive)
- Full path name of the source file or a ~ *user pathname*
- Full path name of the destination name
- User's login name
- A - (hyphen) followed by a **-d** or an **-m**.

Sending Files from Remote Systems

Files are sent from remote systems one of the following ways:

- A **uucp** command sent to the remote system where the **uucico** program runs the command.
- A **uucp -e** command sent to the remote system, requesting that the **uucp** command be executed by the remote system.

  If the **/usr/lib/uucp/QTCMDS** file in the remote system lists **uucp** as a command that can be run remotely, the **uux** program creates and sends the request and the **uuxqt** program runs the command. See "Running Remote Commands (uux)" on page 7-20 and "Remote Commands (QTCMDS)" on page 8-20.
- The source file contains special shell characters (?, *, [, and]).

After the work has been set up in the spool directory, the **uucico** program tries to contact the other computer to perform the work. See "The uucico Command and Problem Determination" on page B-12.

How UUX Works

The following information is for system administrators who need to know how the **uux** command works.

The work of **uux** is carried out by the **uuxqt** program. You do not have direct access to **uuxqt**, but **uux** calls it when needed. See *AIX Operating System Commands Reference* for details about **uuxqt**.

The **uux** command generates an execute file with the names of the files needed to run the command on a remote system and a script that the **uuxqt** program processes.

The **uucico** command assists **uux** by:

- Scanning the spool directory for work
- Placing a call to a remote system
- Negotiating a line protocol to be used
- Running all requests from both systems
- Logging work requests and work completions.

The execute file goes either to the spool directory for local execution or to the remote system with a **send** command.

The files needed to run the **uux** program include:

- Standard output
- System user's login name
- Destination of the standard output
- Command to be run.

If required files are not on the system that is to run the command, **uux** generates **receive** command files which are sent to that system. Then **uucico** runs those files.

The file script consists of several lines, each with an identification character and one or more entries:

user line U *username systemname*

Username and *systemname* are the requester's login name and system.

required file line F *filename realname*

Filename is a unique name used for file transmission and *realname* is the last part of the actual file name and contains no path information. Zero or more of these names can be present. The **uuxqt** program checks for the existence of all these files before running the command.

standard input line I *filename*

The standard input is either specified by a < (less than symbol) in the command string or inherited from the standard input of the **uux** command if the - (dash) option is used.

If standard input is not specified, **/dev/null** is used. If standard input is specified, it also appears in an F (required file) line.

standard output line
O *filename systemname*

Standard output is specified by a > (greater than symbol) within the command string. If a standard output is not specified, **/dev/null** is used. Note that use of > > is not implemented.

command line C *commandstring*

The parameters are those specified in the command string. Standard input and standard output do not appear in this line.

A shell PATH statement appears in front of the command line, as specified in the **uuxqt** program. UUCP checks to see that the command is allowed.

All required files go to the execute directory (usually **/usr/lib/uucp/.XQTDIR**). The shell specified in the **uucp.h** header file runs the AIX command. After execution, the standard output is sent to the requested location.

mail suppression line

Z or N

Z: After the command runs, return a message to the requestor if the command failed (returned a non-zero).

N: After the command runs, do not return a message to the originator.

Chapter 8. Customizing UUCP

CONTENTS

About This Chapter 8-3
Overview of UUCP Customization and Configuration 8-4
Choosing a Name for Your Site (chparm) 8-5
Editing Files 8-6
  Contents of Files 8-6
  File Location of Communication Characteristics 8-7
  How to Edit a File 8-8
  Defining Outgoing Physical Links (L-devices) 8-10
  Phone Number Prefixes (L-dialcodes) 8-11
  System Names (L.sys) 8-12
  Source of Forward Request (ORIGFILE) 8-17
  Next Node (FWDFILE) 8-17
  File Access (USERFILE) 8-18
  Remote Commands (QTCMDS) 8-20
Administrative and Problem Determination Files 8-20
  SQFILE (Sequence Check File) 8-21
  TM (Temporary Data Files) 8-21
  LOG (Log Entry Files) 8-22
  STST (System Status Files) 8-22

About This Chapter

This chapter contains information about customizing the UNIX-to-UNIX Copy Program facility to fit your specific needs. The concepts discussed here require advanced knowledge of computer systems as well as experience with the AIX Operating System and the IBM RT PC.

For examples of customization for modem connections, see "Modem Connect for a Call-Out Port (UUCP)" on page 9-9 and "Modem Connect for a Call-In Port (UUCP)" on page 9-13.

Overview of UUCP Customization and Configuration

If you have installed the Extended Services Program that comes with the AIX Operating System, UUCP is installed. If you have not installed Extended Services, install the UUCP component.

A brief discussion of the installation procedure is in Appendix A. For more detailed information, see *Installing and Customizing the AIX Operating System.*

After UUCP is installed, you need to customize UUCP for your system:

1. Give your site a name, if you have not done this previously.

   See "Choosing a Name for Your Site (chparm)" on page 8-5.

2. Edit several **/usr/lib/uucp** files to add entries that define the communication characteristics:

   - Descriptions of communication links
   - Telephone number prefixes (if needed)
   - Remote systems with which you have direct communication
   - Remote systems for which you forward files
   - Remote systems to which you forward files
   - Files on your system that can be accessed by specific users
   - Commands on your system that can be run remotely.

   See "Editing Files" on page 8-6 for information on editing files.

3. Set up a port by identifying the adapter and communication device you are using, and selecting the device settings that configure the port as call-out or call-in.

   See Chapter 9, "Using Ports, Cables, and Modems."

Choosing a Name for Your Site (chparm)

If you have not selected a site (node) name, choose a unique, lowercase name of seven characters or less. General names like `machineA` or `sys2` are not good choices.

Use the **chparm** command to identify the node name you select. See *AIX Operating System Commands Reference* for additional information.

How to Give Your Site a Name

1. Type **chparm nodename**=*nodename* on the command line.

   To name your site `boston1`, type the following:

   ```
   chparm nodename=boston1
   ```

2. Press the **Enter** key.

3. To have the name take effect, restart the system.

Editing Files

This section lists the contents of each file you can edit and the file location of each communication characteristic you can configure. In addition, an overview of the editing procedure is provided.

Contents of Files

The following is a descriptive list of the files you can edit to add or change communication characteristics:

/usr/lib/uucp/L-devices
: Describes the outgoing physical communication links your system can use. You need this information to call out to other systems.

/usr/lib/uucp/L-dialcodes
: Contains the telephone number prefixes you need for dialing. You need this information to call out to other systems.

/usr/lib/uucp/L.sys
: Describes the remote systems with which your system can communicate. You need this information to call out to other systems.

/usr/lib/uucp/ORIGFILE
: Contains the list of systems and users for which the local system can forward files. This information is needed for forwarding files.

/usr/lib/uucp/FWDFILE
: Contains a list of remote systems to which the local system can forward files. This list is a subset of the **L.sys** file, and is used to limit expensive routes. This information is needed for forwarding files.

/usr/lib/uucp/USERFILE
Contains information on local and remote system users and the files in your system that each user can access. This information on the remote system identifies files you can use.

/usr/lib/uucp/QTCMDS
Lists the commands that remote users can run on your system. This information on the remote system identifies the remote commands you can run.

File Location of Communication Characteristics

The following shows the file location of the communication characteristics you can configure:

Characteristic:	Files to Edit:
bits per second	**L-devices** and **L.sys**
callback	**USERFILE**
commands (remote)	**QTCMDS**
device name	**L-devices** and **L.sys**
dialing code	**L-dialcodes**
dialing device	**L-devices**
dialing prefix	**L-dialcodes**
login name (remote)	**USERFILE**
login sequence (remote)	**L.sys**
name (remote system)	**USERFILE** and **L.sys**

path name	**USERFILE**
phone number	**L.sys**
system name (forward for)	**ORIGFILE**
system name (forward to)	**FWDFILE** and **L.sys**
system name (remote)	**USERFILE** and **L.sys**
telephone number	**L.sys**
time	**L.sys**
transmission speed	**L-devices** and **L.sys**
type of link	**L-devices**

How to Edit a File

To edit a file, follow the procedure described below. The **L-devices** file is used as an example.

How to Edit a uucp File

1. Type the command to enter the editor you are using, after the prompt on the command line.

   For example, to use INed, type the letter **e**.

2. Type the path name after the editor command. If you want to edit the **/usr/lib/uucp/L-devices** file, type:

   ```
   /usr/lib/uucp/L-devices
   ```

3. Press the **Enter** key.

   The file displays.

4. Type the description of each communication device available to your system on a separate line in the file. For the **L-devices** file, use this format:

   ```
   type  device  dialer  speed
   ```

   An example of one line follows:

   ```
   Direct  tty5   tty5   19200
   ```

   See "Defining Outgoing Physical Links (L-devices)" on page 8-10 for a more complete explanation of entries in the **L-devices** file.

5. Press the control key to leave the editor. For INed, press **Alt-D**.

Defining Outgoing Physical Links (L-devices)

If you are not familiar with ports, you may want to read Chapter 9, "Using Ports, Cables, and Modems" on page 9-1 before you read this section.

The **/usr/lib/uucp/L-devices** file defines the communication links available to a call-out port on your system. If your computer only has a call-in port, the file should be empty.

Each line in the file describes one communication link, and has the following format:

```
type  device  dialer  speed
```

An explanation of each field follows:

type Type of connection. Specify one of these:

Direct Line directly cabled to the other system.

Manual Line with a modem, but dialed manually.

ACU Automatic call unit. Call-out line with an auto-dialer attached.

ACU*dialdev*
Call-out line with an autodialer modem of type *dialdev*. See Chapter 9, "Using Ports, Cables, and Modems" on page 9-1 for information on supported modems.

device The name of the device you use to call the remote system. If the device is `/dev/tty1`, place `tty1` in this field.

dialer The device, such as `tty1`, that you use to dial the telephone number. This generally is the same as the *device*.

speed Line speed, measured in bits per second. Also called transmission speed. Acceptable values are 300, 1200, 2400, 4800, 9600, and 19200.

If a line can accommodate more than one speed, type a separate entry for each speed. Be sure to use the same device name for each entry.

The two-entry file below includes an entry for a direct connection and an entry for a modem connection.

```
DIRECT            tty5 tty5 19200
ACUhayes_1200     tty4 tty4 1200
```

Phone Number Prefixes (L-dialcodes)

If your telephone system uses tie lines, long distance access numbers, or other dialing codes, you can define and store the prefixes in the **/usr/lib/uucp/L-dialcodes** file. The telephone number itself is stored in the **L.sys** file. Each dial code entry has the following format:

```
name    number
```

name Short alphabetic name that identifies the phone number prefix.

number The phone number prefix, such as an area code.

For example, if your **L-dialcodes** file has the entry `nyc 9-1-555-`, when you call a phone number in the **L.sys** file called `nyc111-1111`, the number is dialed as `9-1-555-111-1111`.

If you do not use dial codes, specify the full phone number as an entry in the **L.sys** file.

System Names (L.sys)

The **/usr/lib/uucp/L.sys** file stores names and descriptions of remote systems with which the local system can communicate.

Call-Out Sites

You need an entry for each remote phone number or device you can contact. If more than one transmission speed can be used, you need an entry for each speed for a number or device.

Use the following format to describe each contact you can make:

name time device speed phone login

Each entry in the **L.sys** file may be 300 characters long.

See "Sample Entries" on page 8-16 for sample entries.

Call-In Only Sites

If your site can only receive calls (as a secondary or call-in site), you need an entry to identify each remote system that can call you. This entry needs only a *name*, *time*, and *device*:

name time device

If you have a call-out/call-in site, you do not need an entry for incoming calls.

See "Sample Entries" on page 8-16 for a sample entry.

Note: If you need to send a file to a remote system from a call-in site, you can send the file to a queue with the **uucp** command. See "Sending Files" on page 7-12. When the remote system calls in for any reason, the **uucico** program automatically looks for files to deliver. If an entry for that remote system is in your **L.sys** file, the **uucico** program delivers the queued file. Without an entry in the **L.sys** file, your file cannot be delivered, and it is returned to you.

An explanation of each field follows:

name uucp site name of the remote system.

time The *days* of the week and *times* of day when the local system is permitted to call the remote system, or when a remote system can contact a call-in site.

days Any combination of `Mo Tu We Th Fr Sa Su Wk Any`.

`Wk` means any weekday and `Any` means any day.

times Uses the 24-hour clock format. If the field is blank, any time is acceptable. A retry time may be added, using the 24-hour clock.

Since a local system usually accepts incoming calls any time, `Any` is a typical choice for entries describing remote calls you can receive.

If you want to restrict outgoing calls on weekdays to the hours before 9 am (0000-0859) and after 5 pm (1701-2400), with no weekend restrictions, type:

```
Wk0000-0859,Wk1701-2400,Sa,Su,0001
```

The last number `0001` sets the retry interval to one minute, instead of the default retry time.

device Name of the device you use to call the remote system. If you are a call-in only (secondary) site, this is the name you use to identify your site.

If the remote system is directly connected to a device by a cable and does not use a modem, use the device name, such as `tty1`.

If you use a call-out line, *device* must be `ACU`.

If you have a call-in site only, *device* must be `SLAVE`.

speed The transmission (line) speed in bits per second.

phone The phone number to dial. For a direct connection, use *device*, such as `tty1`.

login Login sequence that you need to login and give your password at the remote system. The login name generally is *uucp*. The password is whatever you choose.

The login-sequence entry has two parts, an `expect` string and a `send` string. You must receive the expected prompt before you can successfully send a response (your login or your password).

Since the remote system may not display the prompt that you expect the first time you try, the format gives you several chances to receive the expected string:

```
expect  send  expect  send ...
```

expect The character string you expect from the remote system before you login or before you give your password. This usually is a prompt, such as `login:` or `password:`.

send The character string you send to the remote system. This string can be your login or your password and/or a special value from the following list:

EOT An end-of-transmission character

BREAK A break character

"" A null *send* field.

The *send* field usually has a new-line character at the end.

The following example shows the *login* sequence for dialing a remote PC RT from which you expect to

receive `login:` before you can send your `uucp` login. You also expect `ord:`, (the last part of a password prompt) before you can send your password `bliz:`

```
login:-BREAK-login: uucp ord: bliz
```

The example uses a BREAK character between two attempts to receive the expected string `login:`. If you do not receive the login prompt the first time you try, the system sends a BREAK signal and tries again.

Escape Sequences

The following escape sequences can be used in both the *expect* and the *send* fields:

Symbol	Meaning
\n	New-line (line-feed) character
\r	Carriage return character
\s	Space character
\t	Horizontal tab character
\c	Special indicator that means not to append a new-line character to the field when it is sent. By default, UUCP sends a new-line character after each string.

In the first sample entry below, **\r** is used as a break signal if the expected login does not appear.

Sample Entries

Since call-in sequences, login prompts, user names, and passwords differ from system to system, it is important to use the correct sequence. The examples in this book are for one RT PC communicating with another RT PC.

The following example shows a complete entry in the **L.sys** file of system `xyz` for dialing remote system `ff` with password `bo`:

```
ff Any ACU 1200 5551111 gin:-\r-gin:-\r-gin: uucp ord: bo
```

This says to wait for the remote system `ff` to say `gin` (the last part of the login prompt) before sending `uucp` (the login name). If it does not display the expected `gin`, send a carriage return character and try again; this step is repeated. Wait for `ord:` (the last part of the password prompt) before you send the password `bo` to complete the login at remote system.

The entry in the **L.sys** file of system `ff` (a call-in site) that permits it to receive calls from system `xyz` (now the remote system) is: `xyz Any Slave.`

If the remote system expects the local system to send first, you can put an explicit null in the *expect* field with two double quotes. In the following example, the remote system expects a null string and a carriage return without a new-line before it sends a login prompt:

```
ff Any ACU 1200 5551111 ""\r\c gin:-BREAK-gin: uucp ord: bo
```

Security

Since the **L.sys** file contains sensitive security information, such as login sequences and passwords for remote computers, use at least 0600 permissions and file ownership by *uucp*. General users should not need to examine the **L.sys** file directly, since the **uuname** command gives the names of compatible systems.

Source of Forward Request (ORIGFILE)

The **/usr/lib/uucp**/ORIGFILE contains the names of remote systems and users for which files can be forwarded through your system. Each entry represents the source system, not the last one to forward a file.

Each entry in ORIGFILE has the following format:

systemname![(*username*)]

systemname — Name of a remote system for which the local system can forward a file. Refers to the source system, not the last one to forward a file.

username — Name of a user on *systemname* who can originate a forward request.

Next Node (FWDFILE)

The **/usr/lib/uucp**/FWDFILE contains the names of remote systems to which you can forward a file through your system. These systems must be a subset of **L.sys**, and are needed to limit expensive routes.

Each entry in FWDFILE has the following format:

systemname![(*username*)]

systemname — Name of a remote system to which the local system can forward a file. Also must be listed in the **L.sys** file.

username — Name of a user on *systemname* to which the local system can forward a file.

File Access (USERFILE)

The **/usr/lib/uucp/**USERFILE contains information about the login and system names of compatible remote computers, local and remote users, and the files each user can access.

Each entry in USERFILE has the following format:

login,sys [c] *pathname* [*pathname*]

login	Login name for a user or a remote computer that communicates with the local system.
sys	System name of a remote computer that communicates with the local system.
c	Call-back required. Tells a remote system whether the local system it tries to access will call back to check its identity.
pathname	Path name prefix for files accessible to a general user on the local system or accessible to the remote system.

Permitted Access

The **uucp** program looks for the allowable path name in the following way:

Files Stored and Used on the Local System: The allowable path names are in the first USERFILE entry that contains the login name of the user. If there is no such entry, the program uses the first entry with a *null* login name.

File Access from a Remote System (Call-In Mode): The allowable path names are in the first USERFILE entry that contains the system name matching the remote machine's name. If there is no such entry, the program uses the first entry with a *null* system name.

In the following sample entry, remote computer bon with login name iota requests the transfer of files with path names that start with /usr/mobiz. There is no call-back flag.

```
iota,bon /usr/mobiz
```

File Access by Log-in Name from a Remote Computer: The login name the remote computer uses must appear in the USERFILE. The file may contain several entries with the same login name, but one entry must hold either the name of the remote system or a *null* system name.

Several examples follow:

In this example, any remote computer can login with the name **uucp** since no system name is specified, but it can transfer only those files in the spool directory:

```
uucp, /usr/spool
```

If a file name is added to the example above, that file can be accessed. An example is:

```
uucp, /usr/spool /usr/joy
```

The next example permits any login name from any system to access any file whose name starts with **/usr**:

```
, /usr
```

File Access for a Remote Login with a Call-back Flag: The local system checks the identity of the remote system by calling it back before permitting any transactions. This helps to prevent unauthorized use of the local system.

Note: The owner of a file to be transferred must set the permission bits so that the file is readable by anyone.

Remote Commands (QTCMDS)

The **uux** command can run only those commands listed in the **/usr/lib/uucp/QTCMDS** file. To allow a remote user to run any command on the local system, put Any as the first line in the file. Most systems severely limit remotely executable commands.

The **QTCMDS** file should at least contain the **rmail** command.

Other commands can be added as appropriate, such as **print**, **ls**, **cmp**, and **diff**.

Administrative and Problem Determination Files

The following files do not need to be edited, since they contain administrative data and can be used for problem determination.

SQFILE (Sequence Check File)

The **/usr/lib/uucp/SQFILE** is an administrative file set up in the **uucp** program directory. It contains an entry for each remote system with which you perform conversation sequence checks.

The initial entry is the system name of the remote system. The first conversation adds the conversation count as well as the date and time of the most recent conversation. These items are updated with each conversation. If a sequence check fails, you then need to adjust the entry manually.

TM (Temporary Data Files)

These files are created in the **spool** directory while a file is being copied from a remote machine. The name of a temporary file has the following format:

TM.*pid.ddd*

The *pid* is a process ID and *ddd* is a sequential three-digit number starting at 0. After the spool directory receives the entire file that is being copied into the **TM** file, this file is moved or copied to the requested destination. If processing ends abnormally, this file remains in the spool directory.

You should periodically remove **TM** files with the **uuclean** command. The following line removes all **TM** files older than three days:

```
/usr/lib/uucp/uuclean -pTM
```

LOG (Log Entry Files)

While UUCP is running, log information is added to the **LOGFILE**. If this file is locked by another process, the log information goes in individual log files that have the prefix **LOG**. You can combine these files and add them to information already in the **LOGFILE** with the **uulog** command. See *AIX Operating System Commands Reference* for information on flags to use with this command.

The **LOG.files** are created initially with mode 0222. If the program that creates it ends normally, the mode changes to 0666. If it ends abnormally, it may leave the mode at 0222. If so, **uulog** cannot read or remove the file. To remove the file, use **rm** or **uuclean** or change the mode to 0666 and let **uulog** merge it into the **LOGFILE**.

STST (System Status Files)

The **uucico** program creates these files in the spool directory. They contain information on login, call-out, or sequence check failures or, while two machines are communicating successfully, they contain a **talking** status. The name of each file has the following format, where *sys* is the name of the remote system:

STST.*sys*

For ordinary failures, such as login or dialing, the file prevents another try for a default time of about 55 minutes.

For more serious sequence check failures, you must remove the file before trying again to communicate with that system.

Chapter 9. Using Ports, Cables, and Modems

CONTENTS

About This Chapter 9-3
Ports 9-4
  Types of Ports 9-4
  Using Port Commands (penable, pdisable, phold) 9-5
  How to Set Up a Port (devices) 9-6
Connecting Terminals and Modems 9-7
  Adapters 9-7
  Cables 9-7
  Modem Eliminators (Null Modems) 9-8
Modems 9-9
  Modem Connect for a Call-Out Port (UUCP) 9-9
  Modem Connect for a Call-In Port (UUCP) 9-13
  Switch Settings for Supported Modems 9-16
  Autodialer Programs 9-16
  Sample Autodialer Program 9-18

About This Chapter

This chapter provides an overview of the role of ports in communication, and describes how to set up a port to customize a communication facility for your system.

This chapter also introduces the adapters, cables, and modems you need.

Two examples developed for modem connections show the relationship between this chapter and those aspects of UUCP customization discussed in Chapter 8, "Customizing UUCP":

- Choosing a site name
- Editing several files to define the communication characteristics.

The information on setting up a port described in this chapter is generic, so it applies to any AIX facility that requires a port:

- Identifying the adapter and communication device to your system
- Selecting device settings to create a call-out or a call-in port.

Ports

This section provides an overview of ports and then tells you how to:

- Use port commands (**pdisable**, **penable**, **phold**) to change the port configuration to control logins, so you can call out or set your system to receive incoming calls.
- Set up a port to customize your system for the communication facility you want to use. You do this with the **devices** command.

As used in this guide, a ***port*** is the logical connection between your RT PC and another unit, such as a computer, a printer, or a terminal. A port has programs and files associated with it.

Physically, a port is supported by an adapter in the RT PC, connected to another unit by one of the following:

- A direct cable which permanently connects a device (such as a terminal) to the adapter. This is called ***hardwiring***.
- A modem and common carrier line for communications over short or long distances
- A modem eliminator (null modem), a device that functions like a modem for short distances.

Types of Ports

There are two port states:

Call-out Initiates a connection to a remote system. To do this, the port must be ***disabled***, using the **pdisable** command, so another system cannot log into it. A call-out port is also called a primary site.

Call-in Receives logins from a remote system. Serves as the gateway to the resources of your system. The port must be ***enabled***, using **penable**, to receive a login.

If the only port on a system is a call-in port, the port is called a secondary site.

A port is set up either for a call-out or call-in. Some ports switch functions in response to a port command. For example, in the discussion on UUCP, ports were disabled and enabled, depending on whether a file was sent or received.

Using Port Commands (penable, pdisable, phold)

You can change the function of a port by giving the **penable** or **pdisable** command. These commands change the configuration of the port and make the port available or not for a login. A third command **phold** is a conditional disable command.

penable *portname*
Enables the port. A request for a login can be accepted.

pdisable *portname*
Disables the port. A request for a login cannot be accepted.

phold *portname*
Arranges for a port to be disabled as soon as the current user logs out.

These commands work by changing the status of the port in the **/etc/portstatus** file. This makes the commands temporary rather than permanent. The next time the system is turned on, the default value in the **devices** command is in effect.

You can use the three commands to change a single port or a class of ports. Examples and explanations follow:

- `pdisable term = dialin`

  This line disables all ports configured for calling in.

- `penable speed = 9600`

  This line enables all ports with transmission speeds of 9600 bits per second.

- `pdisable tty0`

  This line disables logins for a device.

How to Set Up a Port (devices)

To set up a port, you must:

- Identify to the system the adapter port and the device you want to connect.
- Select the device settings that create either a call-out or a call-in port.

To start both tasks, use the system **devices** command and select **add** when the Device Customizing Commands menu displays.

As you set up a port, you must know the:

- Device class
- Device type
- Adapter name
- Port number, if requested
- Any changes in device settings (default values) you need to make.

If you change settings later on, disable the device before you type the **devices** command.

Information on devices and the **devices** command is in *Installing and Customizing the AIX Operating System.* See also "Modem

Connect for a Call-Out Port (UUCP)" on page 9-9 and "Modem Connect for a Call-In Port (UUCP)" on page 9-13.

Connecting Terminals and Modems

This section briefly describes the adapters and cables that you need to connect terminals and modems to a system through asynchronous ports.

For more detailed information, see *User Setup Guide* and *Site Planning and Preparation Guide.*

Adapters

You can use one of three serial adapter cards or one of the serial ports on the system board for communication between the local system and a remote one:

- An AT serial/parallel card (for a modem attachment)
- A multi-port RS-232 card (for a modem attachment or a direct connection)
- A multi-port RS-422 card (for a direct connection)
- The 6150 system board for a modem attachment or a direct connection.

Cables

Consult the *Site Planning and Preparation Guide* and *User Setup Guide* for descriptions and drawings of the various cable connections.

Modem Eliminators (Null Modems)

When two devices that both function as DTEs (data terminal equipment) are directly connected, the cable that connects them must transpose send/receive signals. A particular type of device, called a ***modem eliminator*** or ***null modem***, is usually required.

The null modem connects a terminal directly to a computer port through a wired connector with a specific pin arrangement.

Modems

There are two types of modems: internal and external modems. An ***internal modem*** is an adapter that fits into the RT. An ***external modem*** is a box outside the system unit that plugs into a single-port or four-port adapter.

You must configure the modem connection for the specific modem that you use. The instructions that come with the modem provide a description of its characteristics.

Modem Connect for a Call-Out Port (UUCP)

This example shows how to customize and configure the UUCP facility for a call-out port with a Hayes_1200 modem. Other modems may require different settings. See Chapter 8, "Customizing UUCP" on page 8-1 for more information on the files and "How to Set Up a Port (devices)" on page 9-6 for instruction on the **devices** command.

The characteristics of the connection are:

modem	Hayes, 1200 bits per second
local sitename	earth
device name	tty1
remote sitename	moon
phone number	9-1-555-999-9999
login name to remote site	uucp
password	car

Configuring a Modem Connect (Call-Out Port)

1. Type the following on the command line to name your site:

   ```
   chparm nodename=earth
   ```

2. Add this entry to **/usr/lib/uucp/L-devices:**

   ```
   ACUhayes_1200 tty1 tty1 1200
   ```

3. Add this entry to **/usr/lib/uucp/L-dialcodes**:

   ```
   aus 9-1-555-999-
   ```

4. Add this entry, all on one line, to **/usr/lib/uucp/L.sys**:

   ```
   moon WkO800-1700 ACU 1200 aus9999 gin:-BREAK-gin:
   uucp ord: car
   ```

   This line allows calls to site *moon* on weekdays from 8 a.m. to 5 p.m.

5. Add these commands to **/usr/lib/uucp/QTCMDS**:

   rmail
   rnews
   who
   echo
   uumove

   This allows remote UUCP users to run **rmail, rnews, who** and **echo** and allows local UUCP users to run **uumove**.

6. Add device **tty1** by entering the information in the next box.

Adding a Device (Call-Out Port)

1. Type the **device** command on the command line and press the **Enter** key.

   ```
   devices
   ```

2. When the Device Customizing Menu appears, type the **add** option, the device class and the device type on the command line and press the **Enter** key:

   ```
   a ttydev tty
   ```

3. In response to the prompts, select the first 232C, 4-port asynchronous adapter and the first port and press the **Enter** key:

   ```
   rs232c1 1
   ```

4. When prompted about changes in the pre-determined information in the system configuration, type **yes** to display and change the current settings.

5. When the configuration appears, set:

   ```
   ae to false (Automatic enable is false)
   ```

   Do not allow people to login through this port.

6. Press the **Enter** key.

7. Indicate that you want to change another option.

8. Set the options as listed in the next box.

Setting Options for Call-Out Port

1. Set the options in the above example as follows:

```
bpc   8 (data sent 8 bits per second)
pt    none  (no parity)
rts   1200 (receive and transmit speed 1200 bps)
sns   true (transmission by modem)
aa    false (no automatic answer for call-out)
dvam  1 (device attached by modem)
nosb  1 (data has one stop bit)
om    full (mode of operation is duplex)
pro   dtr (protocol is dtr--data terminal ready)
ixp   true (do not include Xon/Xoff protocol)
```

2. Use the default for all other options.

3. Press the **Enter** key twice.

If you change a port rather than create one from scratch, enter **penable** and then **pdisable** to put the new settings for the call-out port into effect.

Modem Connect for a Call-In Port (UUCP)

This example shows how to customize and configure the UUCP facility for a call-in port with a Hayes _1200 modem and an RS-232 card. Other devices may require different settings. See Chapter 8, "Customizing UUCP" on page 8-1 for more information on the files and "How to Set Up a Port (devices)" on page 9-6 for an overview of the **devices** command.

The characteristics of the connection are:

modem	Hayes 1200 baud
local sitename	earth
device name	tty1
login name to local site	uucp
remote sitename	moon

Configuring a Modem Connect (Call-In Port)

1. Type the following on the command line to name your site:

   ```
   chparm nodename=moon
   ```

2. Add the following to **/usr/lib/uucp/L-devices**:

   ```
   ACUhayes_1200 tty1 tty1 1200
   ```

3. Add the following to **usr/lib/uucp/USERFILE**:

   ```
   uucp,moon /usr/spool /tmp
   ```

   This allows remote system `moon` UUCP access only to files with pathnames **/usr/spool** and **/tmp**.

4. Add device **tty1** by entering the information in the next box.

Adding a Device (Call-In Port)

1. Type the **device** command on the command line and press the **Enter** key:

   ```
   devices
   ```

2. When the Device Customizing Menu appears, type the the **add** option, the device class and the device type and press the **Enter** key:

   ```
   a ttydev tty
   ```

3. In response to the prompts, select the first 232C, 4-port asynchronous adapter and the first port and press the **Enter** key:

   ```
   rs232c1 1
   ```

4. When prompted about changes in the pre-determined information in the system configuration, type **yes** to display and change the current settings.

5. When the configuration appears, set:

   ```
   ae to true (Automatic enable is true.)
   ```

   Allow people to login through this port.

6. Press the **Enter** key.

7. Indicate that you want to change another option.

8. Set the options as listed in the next box.

Setting Options for Call-In Port

1. Set:

   ```
   aa    true (no automatic answer for call-out)
   ```

   All other settings are the same as for the call-out port.

2. Press the **Enter** key twice.

3. Type `penable tty1` to complete the set up for the call-in port.

The steps described for configuring a call-in port are an example of the general procedure. You may need to set other parameters to match those of the remote system, such as transmission speed, parity, and bit length/character.

Switch Settings for Supported Modems

Hayes Smartmodem 1200

In the settings that follow, up means open and down means closed.

1. Up. DTR (yes/no)
2. Down. Result code (digits/words)
3. Up. Result code (yes/no)
4. Down. Echo (yes/no)
5. Up.
6. Up. Autoanswer (yes/no)
7. Up if using standard RJ11 phone jack; down if using multiline phone system. Carrier detect (yes/no)
8. Down. Command recognition (enabled/disabled)

Configure the associated terminal port with modem control.

Autodialer Programs

Many brands of modems can dial phone calls, but each type requires its own commands to initiate the dialing. Special autodialer programs handle these details. The system provides several, and you can also write new autodialer programs to interface with new kinds of modems as they are developed.

Invoking Autodialer Programs

All autodialer programs are in the file **/usr/lib/INnet/dialers**. The dialer for a Hayes Smartmodem 1200[1], for example, is called **/usr/lib/INnet/dialers/hayes_1200**.

When an autodialer program runs, the *tty* port will already have been opened. The caller passes the program the phone number to dial and, optionally, the name of the dialer hardware to use, if it is different from the modem itself. If the call succeeds, the program exits with a return code of 0. If it fails, it returns one of the following codes:

Return Codes

1	Cannot open dialer.
2	Busy or no answer.
3	Unable to work.
4	Stopped before completion.
5	Communication failure.
6	Busy.
7	No answer.
8	Phone not working.
9	Incorrect phone number.
10	Cannot open device.

The values of 8, 9, and 10 indicate unrecoverable errors.

---

1 Trademark of Hayes Microcomputer Products, Inc.

Phone Numbers

Many autodialer programs use the following conventions in interpreting telephone numbers. If you write your own dialer, you may want to use them:

<space>	Ignored. (Spaces may improve readability of output).
(	Ignored.
)	Ignored.
-	Ignored.
T	Tone dial.
,	5 second delay.

Sample Autodialer Program

On the following pages is a sample C program for an autodialer. You can use it as a model for writing your own autodialer programs.

```
#
/*
 *  Hayes Smartmodem 1200
 *
 *  Makes a phone call using the Hayes autodialer
 *
/*

static char sccsid [ ] = "@(#)
hayes_1200.c   5.3 - 84/07/13";

#include <signal.h>
#include <IN/standard.h>
#include <termio.h>

#define PORT        3


main (argc, argv)
int argc;
char *argv [ ];
{    register char *p;
     int state;
     char numbuf [ 40 ];

     static struct termio hmodes;


     static char cmd0 [ ] = "ATQ0\r";
     static char cmd1 [ ] = "ATE0\r";
     static char *status [ ] =
          { "OK\r\n",
               "CONNECT\r\n",
               "RING\r\n",
               "NO CARRIER\r\n",
               "ERROR\r\n",
               CONNECT 1200\r\n",
               0};
```

```
if (!CSnil (argv [ 1 ]))
{    p = CScpy (numbuf, "ATD ");
     p = CScpy (p, argv [ 1 ]);
     p = CScpy (p, " \ r");

     ioctl (PORT, TCGETA, &hmodes);
     hmodes.c_iflag = ISTRIP | IGNBRK;
     hmodes.c_oflag = 0;
     hmodes.c_cflag &= CBAUD;
     hmodes.c_cflag |= CS8 | CREAD | HUPCL;
     hmodes.c_lflag = 0;
     hmodes.c_cc [VMIN] = 1;
     ioctl (PORT, TCSETAF, &hmodes);

     state = 0;
     for (;;)
     {   switch (state)
         {   case 0:
                    write (PORT, cmd0, (sizeof cmd0)-1);
                    ++state;
                    break;
               case 1:
                    write (PORT, cmd1, (sizeof cmd1)-1);
                    ++state;
                    break;
               case 2:
                    write (PORT, numbuf, p-numbuf);
                    ++state;
                    break;
               case 3:
                    exit(9);
          }
          switch (reply (status, 60))
          {   case -1:
                    exit(5);
               case 0:
                    break;
```

```
                    case 2:
                    case 4:
                         exit(9);
                    case 3:
                         exit(2);
                    case 1:

                         hmodes.c_cflag &= ~CBAUD;
                         hmodes.c_cflag |= B300;
                         ioctl (PORT, TCSETAF, &hmodes);

                         exit(0);
                    case 5:

                         hmodes.c_cflag &= ~CBAUD;
                         hmodes.c_cflag |= B1200;
                         ioctl (PORT, TCSETAF, &hmodes);

                         exit(0);
                    }
               }
          }
          exit(0);
}


reply (possible, timer)
char *possible [ ];
int timer;
{    register char *p;
     register char **s;
     register int n;
     int timeout ( );
     char buf [ 64 ];

     p = buf;
     signal (SIGALRM, timeout);
     for (;;)
```

```
        {    alarm (timer);
             n = read (PORT, p, 1);
             alarm (0);
             if (n < 0)
             {    signal (SIGALRM, SIG_IGN);
                  return(-1);
             }
             if (*p) {

                  *++p = '\0';
                  if (p >= buf + 63)
                      p = CScpy (buf, buf + 32);
                  for (s = possible; *s; s++)
                  {   if (CSlocs (buf, *s) != p)
                      {    signal (SIGALRM, SIG_IGN);
                           return (s-possible);
                          }
                      }
                  }
             }
}

timeout ()
{
          signal (SIGALRM, SIG_IGN);
}



/* @ (#)CScpy.c 5.1 - 84/03/14 */

char *
CScpy(dst, src)
     register char *dst, *src; {

     if (dst && src) {
          while (*dist++ = *src++)
                 ;
```

```
            --dst;
        }
        return dst;
}


/* @(#)CSlocs.c 5.1 - 84/03/14 */

char *
CSlocs(str, pat)
        register char *str, *pat; {
        register char *strp, *patp;

        if (str)
            while (*str) {
                if (CScmpp(pat, str) == 0)
                    break;
            ++str;
            }
        return str;
}
```

Appendix A. Installing Communications Facilities

CONTENTS

About This Appendix A-3
Installing the AIX Operating System A-4
Installing the Communication Facilities A-5

About This Appendix

This appendix contains a brief description of the installation procedure that must be completed before you can use the communication facilities discussed in this book.

As a prerequisite task, the AIX Operating System must be installed if not done previously.

If AIX already is installed, you may need to adjust the minidisk space to accommodate the communication facilities.

For information about installation and minidisk space, refer to *Installing and Customizing the AIX Operating System* and *Managing the AIX Operating System.*

Installing the AIX Operating System

The AIX Operating System includes the following:

Virtual Resources Manager (VRM)
Manages the hardware and software resources of your system.

Base System Program
Contains the basic file system--four file systems on four minidisks plus a dump minidisk used if a system failure occurs. The four file systems are:

/ (root)
/usr
/u
/tmp

To install AIX:

1. Install the VRM.
2. Plan the minidisk space requirements for all the programs you want to install. The plan should include the requirements for the VRM, Base System Program, and other AIX programs, such as the specific communication facilities you decide to use.

   Worksheets and minidisk space requirements for each program are in *Installing and Customizing the AIX Operating System.*

   As you plan the space requirements, remember that two of the facilities, Inter-workstation commands and UUCP support, are components of larger programs. If you do not want to install the entire program, plan space only for the components you want.
3. Install the Base System Program.

Detailed information is in *Installing and Customizing the AIX Operating System.*

Installing the Communication Facilities

This section contains a general installation procedure for installing any of the following:

- Inter-workstation Commands, a component of Multi-User Services
- Asynchronous Terminal Emulation (ATE)
- UNIX-to-UNIX Copy Program facility, a component of Extended Services
- Transmission Control Program for use with the Interface Program (TCP/IP).

Use this general procedure for each communication facility you install.

You can install a single component of Multi-User Services or Extended Services. If you are primarily interested in communication facilities, you can save installation time and minidisk space by installing Inter-workstation Commands and UUCP without including the other facilities on the Multi-User Services and Extended Services diskettes.

If you want to install just the Inter-workstation Commands component of Multi-User Services, select item 3 from the install menu on diskette 1 of Multi-User Services.

If you want to install just the UUCP component of Multi-User Services, select item 6 from the install menu on diskette 1 of Multi-User Services.

To Install a Communication Facility

1. Make sure that no one else is using the system and that no user programs are running.

   To do this, you can stop the system, using the **shutdown** command, and IPL the system again.

2. Log in as superuser or as a member of the system group.

   You may log in as **root**.

3. Type `installp` after the prompt on the command line.

4. Press the **Enter** key.

   You are prompted to insert the first diskette.

5. Insert the first diskette for the communication facility you want to install into the top diskette drive.

   This will be for one of the programs listed on the previous page.

6. Follow the prompts to install the program.

   If a program has components that can be installed separately, such as Inter-workstation Commands or UUCP, the install menu will let you select just that component.

   A message appears when all the files are restored and, a few minutes later, when installation is complete.

7. Remove and store the diskette.

Appendix B. Problem Determination

CONTENTS

Asynchronous Terminal Emulation Problems B-3
Establishing a Connection B-3
During a Connection B-6
Using the Pacing Protocol B-8
Using the Xmodem Protocol B-8
UUCP Problems B-10
Problem Determination with the uustat and uucico Commands B-10
Diagnosing a Port Failure Problem B-17
Tracing Port Failures B-18
Device Drivers and Possible Transmission Problems B-20
High-Speed Lines B-21

Asynchronous Terminal Emulation Problems

Problems may occur when you are trying to establish a connection or while you are using a connection. During a connection, you may experience problems with either pacing or xmodem. Following are several problem areas and steps to take to correct the trouble.

Establishing a Connection

If you cannot dial, check the following:

- Make sure your modem is turned on.
- Make sure the modem switches are set correctly. Refer to the instruction book for your modem.
- Sometimes a previous connection leaves a modem in a state from which it cannot dial. Try turning the modem off and then back on again to reset all the default values in the modem.
- Make sure the modem cable is connected at both ends, into the modem and into the computer. Sometimes the cable can work loose.
- Make sure the computer end of the cable is plugged into the correct slot on the RS-232 card for the port you are trying to open. For more on RS-232 and RS-422 cards, see *User Setup Guide and Options Installation.*
- Check that the cable connecting the computer to the modem is the right type. For more information, see *Planning Guide* or *User Setup Guide and Options Installation.*
- A modem must receive characters at a certain speed in order to dial. For instance, a modem may dial only when it receives characters at 75, 150, 300, or 1200 bps. Check the **rate** setting in the Alter Menu to make sure it is set at a valid speed.

- Check that the dialing prefix in the Alter Menu is correct for your modem. Hayes[1] compatible modems usually use a prefix of ATDT or ATDP, in uppercase. Make sure they are not in lowercase.

- Check that the dialing suffix in the Alter Menu is correct for your modem. You may not need one at all; if you use one, it could reset your modem to a state that does not allow your system to communicate.

- Make sure the phone line is plugged into the modem and that the phone line is active. Try dialing a number manually on the line. If you cannot dial out manually, call the phone company.

If you cannot open a port, check the following:

- Make sure the port you are trying to use is listed in the **/dev** directory. You can check this by keying in the following:

  ```
  li -l /dev/*tty*
  ```

  If the port you want to use is not there, you must create it, using the **devices** command. See "How to Set Up a Port (devices)" on page 9-6, "Modem Connect for a Call-Out Port (UUCP)" on page 9-9, and "Modem Connect for a Call-In Port (UUCP)" on page 9-13.,

- If the port exists, it may not be defined correctly for a modem. Use the **devices** command to check and reset the parameters. Some of the settings you may need to change are the following:

---

1 Trademark of Hayes Microcomputer Products, Inc.

Name	*Value*
dvam (Device Attachment Method)	1 (remote)
bpc (Bits per Character)	5, 6, 7, or 8
pt (Parity Type)	none, odd, or even
ixp (Include Xon/Xoff Protocol)	true
rts (Receive/Transmit Speed)	300 or 1200 for modems
sns (Switched/Nonswitched)	true
tt (Terminal Type)	dumb, VT100, etc.
nosb (Number of Stop Bits)	1, 1.5, or 2

If you change port settings, you must disable the port with the **pdisable** command before you use the **devices** command. After you leave **devices**, issue **penable** and then **pdisable** to put the new settings into effect on the port.

- If you are using a port to call out, you must disable it with **pdisable**. You can check its status by issuing the **penable** command. Any port listed is enabled. To disable one, use the **pdisable** command followed by the port name. For more information on call-out ports, see "Ports" on page 9-4.

  Example:

  ```
  pdisable tty0
  ```

If the other side does not answer, check the following:

- Make sure that the remote modem is turned on, that its switches are set correctly, that the phone line is connected, and that the modem cable is plugged in correctly at both the modem and computer ends.

- Check that the remote modem is in auto answer mode. You can set this by a switch on the modem or by typing a command to a smart modem from the remote keyboard.

- The remote port must be enabled or have another communications package running on it if it does not have the same operating system that the local system has.

- Verify that the number you are dialing is correct.

During a Connection

After you have successfully established a connection, you may have problems while you are using it. Following are several problems and possible solutions.

If garbage or nonsense appears on your screen, check the following:

- Try changing the communication parameters in the Alter Menu. The **length**, **stop**, **parity**, and **rate** parameters must all match what is being used on the other end.

- It is possible in AIX for two system users to open the same port. When this happens, both users are trying to read the port at the same time. One user gets some of the characters, and the other gets the rest. One user must stop trying to use the port. To do this, press **Ctrl-R** to disconnect, or press **Ctrl-V** to get the Connected Main Menu and then select **terminate**.

- When you are using a port to call out, it must be disabled. If someone else enables it during your connection, the send and receive lights on your modem appear to be in constant use and the program does not appear to work. In this case, press **Ctrl-R** to disconnect.

If you are losing characters, check the following:

- At high transfer speeds (9600 and 19200 bps), you may find that you are losing characters or getting buffer overflow messages. This may happen when the computer is very busy and several

programs are running at once. You can continue to use these speeds if the remote computer supports Xon/Xoff protocol. Use the Modify Menu to set Xon/Xoff to on, and continue with your connection session.

If control keys do not work, check the following:

- Since control keys do not display on the screen or print on paper, this problem may be difficult to trace. The main symptom is that when you press one of the control key sequences (**Ctrl-R**, **Ctrl-B**, or **Ctrl-V**), nothing happens. Look in the **ate.def** file to see if the control keys have been reset. If so, you can delete the file, and the next time you run Asynchronous Terminal Emulation, the program creates a new file, using the program defaults.

- Your terminal may use a different control key combination to produce the code recognized by the program. The program defaults are the following:

  Ctrl-B 002 octal, 0x02 hex, STX ASCII

  Ctrl-R 002 octal, 0x12 hex, DC2 ASCII

  Ctrl-V 026 octal, 0x16 hex, SYN ASCII

  Check your terminal book to see which control key combinations produce the codes you need.

If your modem does not hang up, check the following:

- You may need to reset the modem if you were disconnected abruptly. Try turning the modem off and then back on.

- There may be too many processes using the port. You can check this by issuing the shell command **ps -el** and looking for processes that should not be running. If you find any, you can eliminate them by using the shell command **kill -9** *PID*, where PID is the process identification number obtained from the **ps** command.

When you are using Asynchronous Terminal Emulation, one or two processes may be running:

- **ate** runs all the time you are using the program
- **atefork** runs when you have an active connection.

On multiple user systems, more than one person may be running the program at a given time. Make sure you eliminate only processes that should not be running.

Using the Pacing Protocol

If a file does not transfer, check the following:

- If you receive message 062-002 while using either interval or character pacing to receive a file at the local site, possibly the remote program did not send an EOT character. Check the file you received to make sure all the data arrived. If so, no further action is necessary. If not, you may want to retry.
- Files sent or received using the pacing protocol should not contain embedded control characters. If they do, the results are unpredictable. Files should contain only text.

Using the Xmodem Protocol

If linefeed/carriage return characters are not correct, check the following:

- UNIX files traditionally contain only linefeeds (called newlines) to delimit the end of lines. DOS or CP/M files usually use a carriage return/linefeed combination. After you transmit the file, if the linefeed/carriage returns do not look right, you can use one of the two following AIX procedures on the file either before you send it or after you receive it:
  - BEFORE sending from AIX to DOS or CP/M, key in the following line to change the newlines to a combination of carriage return/linefeeds:

```
sed 's/$/' 'echo '\r\c''/ file1 > file2
```

The *file1* field contains the original name of the file, and the *file2* field contains a new name for the file. Send the modified file, *file2*.

Note: See the **sh** command in *AIX Operating System Commands Reference* for other quoting mechanisms in command substitution.

- AFTER receiving from DOS or CP/M to AIX, key in the following two lines to remove the carriage returns from the file:

```
tr -d '\015' < file1 > file2
```

```
mv file2 file1
```

If you get garbage or nonsense on the screen, check the following:

- When you interrupt an **xmodem receive** with **Ctrl-R**, the **xmodem** shell command continues to send. The characters it sends appear on the connection screen. This is normal. Press **Ctrl-X** to end the **xmodem** shell command and continue with your connection screen.

UUCP Problems

Problem Determination with the uustat and uucico Commands

You can use these two commands to gather information about problems that may occur in using the UUCP facility. These commands can help you debug the files, directories, and programs that UUCP uses.

The uustat Command

With this command, you can display the status of previous **uucp commands** or connections with other systems. You can also cancel a **uucp command**.

By using one or more of the flags available with this command, you can get information on the status of specific UUCP jobs, systems, and users, over various time periods.

Flag	*Result*
-m*mch*	Report accessibility status of machine *mch*. If you specify *all*, you get the status of all machines known to the local UUCP facility.
-k*jobno*	End the UUCP job number *jobno*. You must own that request or else be the superuser to use this option.
-c*hour*	Remove the status entries older than *hour* hours. This administrative option can be initiated only by the user *uucp* or by the superuser.
-u*user*	Report the status of all UUCP requests issued by *user*.
-s*sys*	Report the status of all UUCP requests for remote system *sys*.

-o*hour* Report the status of all UUCP requests older than *hour* hours.

-y*hour* Report the status of all UUCP requests younger than *hour* hours.

-j*jobno* Report the status of the job you specified. To see the status of all the UUCP requests, use `all` as the job number.

-v Report the UUCP status verbosely. See the following list for return codes and explanations.

Note: Options **-j**, **-m**, **-k**, and **-c** are mutually exclusive and cannot be combined. Use one of them either alone or with one or more of the other flags. If you do not specify an option, **uustat** gives you the status of all the UUCP requests issued by the current user.

When you key in `uustat -v`, you get a descriptive report on the **uucp command** status. If you do not specify this option, you get only the numbers. The list of numbers and descriptions follows:

Octal	Status
000001	The copy failed but the reason is not known.
000002	Permission to access local file is denied.
000004	Permission to access remote file is denied.
000010	Bad **uucp command** is generated.
000020	Remote system can not create temporary file.
000040	Can not copy to remote directory.
000100	Can not copy to local directory.
000200	Local system can not create temporary file.
000400	Can not run **uucp command**.

001000	Copy succeeded.
002000	Copy finished and job is deleted.
004000	Job is queued.
010000	Job is halted (incomplete).
020000	Job is halted (complete).

The uucico Command

This command is called by the **uucp command** to perform the work of transferring files. It does the following five tasks:

1. Scans the spool directory for a job. In the spool directory, the names of work-related files have the following format:

   type . system_name grade number

Field	Description
type	An uppercase letter • C = copy command file (work file) • D = data file • X = execute file.
System name	Name of the remote system
Grade	A character indicating priority
Number	A four-digit, zero-padded sequence number

   For example, the file `C.lab9n0031` is a work file for a transfer between the local machine and the `lab9` machine.

   The **uucico** command scans the spool directory looking for work files (those with a prefix of C) and makes a list of all

systems to be called. The program calls each system and processes the work files.

2. Places a call to a remote system.

   The **uucico** program uses information in several files in the UUCP directory, usually **/usr/lib/uucp**. At the start of the calling process, the program sets a lock to prevent multiple conversations between the same two systems. See "Cleaning up Lock Files" on page B-16.

   The program uses the information in the **L.sys** file to make the call and also checks the **L-devices** file to find an available device for the call. It creates a lock file if a device successfully opens.

   The conversation between the local and the remote **uucico** programs starts with a handshake initiated by the called or secondary system. It lets the calling or primary system know that it is ready to receive the system identification and conversation sequence number. The primary system sends these, the secondary system verifies the information, and protocol selection begins. If the secondary system responds with a *call-back required* message, the current conversation ends and the primary system calls later.

3. Negotiates a line protocol to be used.

   The secondary system sends a message containing a string of characters, each representing a line protocol. The message has this format:

   P*proto-list*

   The calling system checks the list for a letter corresponding to an available line protocol and returns a message telling the secondary system which one to use. The message has the following format:

   U*code*

If the two systems have no common protocol, the *code* field contains **N**.

4. Carries out work requests from both systems.

   The primary system searches for work requests from the secondary system and sends appropriate messages during the processing: **S** for Send a file, **R** for Receive a file, **C** for Copy complete, **X** for Execute a UUCP command, and **H** for Hangup. The primary system sends R, S, or X messages until all work for the remote system is finished and then sends H to end the conversation.

   The secondary system responds with SY, SN, RY, RN, XY, XN, HY, or HN, corresponding to *yes* or *no* for each request.

   The secondary system's S and R replies are based on permission to access the requested file or directory. The permissions come from the USERFILE and from the read and write permissions of the requested file or directory. After each file is copied into the spool directory of the receiving system, the receiver of the file sends a C (copy complete) message. It sends CY if the file has successfully moved from the spool directory to the destination. Otherwise, it sends CN, and the file is put in the public directory, usually **/usr/spool/uucppublic**, and the requester gets a mail notice.

   The response to the H message is determined by a scan of the secondary system's spool directory. If work for the primary system exists, the secondary system sends HN, and the two systems switch roles. If no work exists, HY is sent.

5. Logs job requests and completions.

   Both systems log the requests and the results. See "LOG (Log Entry Files)" on page 8-22.

6. Ends the conversation.

   When the primary system receives HY and the secondary one echoes it back, both systems turn off the protocols and send a final oo (over and out) to each other.

Generally, another program (**uucp**, **uux**, **uuxqt**, or a prior request to **uucico**) or a system daemon starts **uucico** running. You also have direct access to it, usually for testing purposes.

Two of the command's three flags are useful for finding problems in UUCP. When you key in `uucico -d`*directory*, with an actual directory name in the *directory* field, the command uses the specified directory for storing files, instead of the spool directory. This feature gives you a safe place to examine the first step of the UUCP mechanism before the files are in a position to be moved to a remote system.

When you key in `uucico -x`*num*, with a number between 1 and 9 in the *num* field, you get additional information. The higher the number, the more information you get. For example, entering the command `/usr/lib/uucp/uucico -r1 -sbravo -x9` may produce the following type of output. Only the first few lines are shown here, but you usually get much more:

```
**START**
UID  user zomix
uucico -x9 -sbravo
finds called
getto called
call no.8385530< for sys bravo dial 8385530<
suspend tty0
dc - /dev/tty0, acu -/dev/tty0
ACU write ok
dcf is 6
dcr returned as 6
login called
wanted "" got that
wanted gin: login: got that
wanted ord: \40\15\121login:
      \40uucp\15\12Password: got that
```

If you have trouble logging in to a remote system, and most people do at first, you can use the **uucico** command to tell UUCP to

display at your terminal a transcript of the login attempt. Following is a series of commands to help you do this:

```
cd /usr/spool/uucp
rm -f STST.*  #(Status files, useless while tracing errors)
/usr/lib/uucp/uucico -r1 -sremote_sys -x9
```

The flags used here with **uucico** are the following:

- `-r1` Sets the local system to primary mode.
- `-s` Says to call the remote system named after `-s`.
- `-x9` Gives the most tracing information, `-x1 the least`.

Using the UUCP ERRLOG File

UUCP creates this file in the spool directory to record UUCP system errors, which should not happen often. The messages come from the ASSERT statements in the various programs. Wrong modes on files or directories, missing files, read/write system call failures on the transmission channel, and improper configuration in the files can cause entries in the **/usr/spool/uucp/ERRLOG** file.

Cleaning up Lock Files

UUCP creates a *lock file* for each device in use and each system conversing. This action prevents duplicate conversations and multiple attempts to use the same device. The form of the system lock file is **LCK..***sys*, where *sys* is a system name. Device lock files have the form **/etc/locks/***dev*, where *dev* is the simple name of the device. You can leave the files in the spool directory if UUCP fails, for example if the system goes down. The files are removed after 1.5 hours. If you do not want to wait that long before retrying, remove the lock files.

Diagnosing a Port Failure Problem

Hardware problems in a port failure may occur because of difficulties in a piece of communications equipment rather than in the port itself. You may need the help of the telephone company or communications services and the equipment vendor.

Sometimes the problem is line noise. Terminals may pick up noise if they are not properly terminated. The modem may be connecting and then immediately disconnecting many times a minute. As soon as a line becomes active, it immediately goes inactive, sending a hangup signal to the process that was waiting for the line. The process halts and **init** starts another one to take its place. Since this action may happen many times a minute, it severely degrades the performance of the system. The system initialization process monitors the activity on each line to prevent this from happening. If a line goes through a certain number of loggers in a specified period of time, an error message goes to the operator's console and the line is disabled for a few minutes. You should investigate any line that is disabled often.

Other problems may include a port that has not been properly closed and made available to someone else because the previous system user left a process running instead of completing the hang up operation. You can find these problems with the **ps** command and then get rid of them with the **kill** command.

Also, if the special file for a port can not be opened, the **logger** program will end as quickly as it begins. Check the special file to make sure its values are correct. When you issue an **open** call on a communications port, it will wait until it receives a carrier detect signal.

Another way to approach problems is to use the **-d** flag with the **getty** program. This helps you find errors in the device configuration. If you invoke `getty -d`, it does not actually run the logger or other program it is supposed to run. Instead, it displays on your screen the name of the logger or other program it would have run. You can then see if the name is the one you expect.

Also see "Tracing Port Failures" on page B-18.

Tracing Port Failures

In addition to the following hardware diagnostics, you may want to refer to material on using **getty -d** in "Diagnosing a Port Failure Problem" on page B-17.

If you have a line failure problem, take the following steps:

1. Check the obvious:
   - Make sure the terminal is on line.
   - See that the cables are connected.
   - Verify that the transmission speeds match.
2. Connect a different terminal and modem.
   - If a dial-in line is causing a problem for one person, see if others are having the same difficulty.
   - If not, first try to dial the modem where the problem is occurring. Swap modems if you cannot dial it; swap terminals if you can dial it.
3. Determine whether more than one port has failed, especially contiguous ones.
   - If you are using a rotary or distribution panel, check the rotary if dial-in lines are connected to adjacent ports that have failed.
4. Determine whether the problem lies on the computer side or the terminal side of the distribution panel. At the distribution panel, switch a line that does not work with a line of the same type that does work.
   - If the problem moves to the new line, there is probably a hardware or software problem within the system.

- First, check the hardware again and make sure all cables are tight. If the problem still exists, find out what in the system has changed recently, and focus on that area.

- Then check the software. Make sure the special file entry for the port in the **/dev** directory is still intact. Then try to use the port as an output device by using the **cat** command to read a text file and direct its output to the port's special file. If this attempt is successful, the problem is very likely in the software. If the attempt fails, the problem is very likely an electronic one.

• If the problem remains on the same line where it first occurred, there is something wrong with the wire or the equipment along its length.

  - If you have not already tested the terminal, you should do so now.

  - Do the same for the modem by swapping in an identical unit.

  - Finally, look for the problem in the phone company's network, in the DAA (Data Access Arrangement), or in a dial-in modem by means of the following symptoms:

    — If someone tries to dial in but gets no ring or busy signal, the problem may lie in the phone company's network.

    — If the person gets a busy signal but verifies that not all the lines are in use, first switch modems and DAAs or lines. If the problem moves with the modem, get a new unit.

    — If the line rings but does not answer, the problem lies in the modem or in the DAA. But check the ring indicator on the modem to make sure the correct line is in fact ringing.

- If the line rings and answers but no carrier comes up, then most likely either the dial-in or the dial-out modem is at fault.

- If the line answers and the carrier comes up but you do not receive a prompt or the system ignores your input, your terminal or modem may be causing the problem.

Device Drivers and Possible Transmission Problems

After you have completed the cabling, you should test for problems. A device driver within the system controls all ports and determines many characteristics of the terminals on the system. For example, a terminal does not generate a DTR (data terminal ready) signal to a modem until there is an **open** system call pending on the associated port. A modem connected to that port does not answer incoming calls unless a process is trying to open the port. If the port is not open, input on the line is lost.

Once the modem answers and a connection is established, the modem must then generate DCD (carrier detect) for the **open** call to complete. An **open** call that never returns a value may be due to the modem's failure to generate the DCD signal.

After an **open** call completes, you may still send or receive data incorrectly because of mismatch problems in speed, parity, or character length. You can check and change the parameters associated with the port by using the **devices** command. The *AIX Operating System Commands Reference* and *Installing and Customizing the AIX Operating System* have additional information.

If a modem connection is broken but the program does not detect the break, you should check the minor device number in the special file to make sure that modem control is enabled. Some modems have switches or jumpers to configure so that if the connection breaks, the DCD (carrier detect) signal also fails.

High-Speed Lines

High-speed lines (over 2400 bits per second) may cause problems. For example, RS-232C was designed for high-speed transmission over distances of no more than 50 feet.

Refer to *Planning Guide* for further discussion.

Glossary

address. (1) A name, label, or number identifying a location in storage, a device in a network, or any other data source. (2) A number that identifies the location of data in memory.

addressing. (1) In data communications, the way that the sending or control station selects the station to which it is sending data. (2) A means of identifying storage locations.

American National Standard Code for Information Interchange (ASCII). The code developed by ANSI for information interchange among data processing systems, data communications systems, and associated equipment. The ASCII character set consists of 7-bit control characters and symbolic characters.

argument. An expression that is passed to a function, subroutine, or procedure for evaluation.

asynchronous transmission. In data communications, a method of transmission in which the bits included in a character or block of characters occur during a specific time interval. However, the start of each character or block of characters can occur at any time during this interval. Contrast with *synchronous transmission.*

authorize. To grant to a user the right to communicate with, or make use of, a computer system or display station.

auto-call unit. A common carrier device that allows IBM RT PC to automatically call a remote location.

background process. (1) An activity that does not require operator intervention that can be run by the computer while the work station is used to do other work. (2) A mode of program execution in which the shell does not wait for program completion before prompting the user for another command.

bad block. A portion of a disk that can never be used reliably.

basic addressable unit (BAU). The smallest piece of storage that can be addressed. On the IBM RT PC, a byte is the BAU.

batch compilation. A method of compiling programs without the continual attention of an operator, as a background process.

batch processing. A processing method in which a program or programs process records with little or no operator action. This is a background process. Contrast with *interactive processing.*

binary synchronous communications (BSC). A form of communications line control using transmission control characters to control the transfer of data over a communications line.

block. (1) A group of records that is recorded or processed as a unit. Same as *physical record*. (2) Ten sectors (2560 bytes) of disk storage. (3) In data communications, a group of records that is recorded, processed, or sent as a unit.

bps. Bits per second.

breakpoint. A place in a computer program, usually specified by an instruction, where execution may be interrupted by external intervention or by a monitor program.

BSC. See *binary synchronous communications*.

buffer. (1) A temporary storage unit, especially one that accepts information at one rate and delivers it at another rate. (2) An area of storage, temporarily reserved for performing input or output, into which data is read, or from which data is written.

bus. One or more conductors used for transmitting signals or power.

carrier. A continuous frequency that can be modulated with a second (information-carrying) signal.

carrier return. (1) In text data, the action causing line ending formatting to be performed at the current cursor location followed by a line advance of the cursor. Equivalent to the carriage return of a typewriter. A keystroke usually indicating the end of a command line. This is generally accomplished by pressing the Enter key.

character string. A sequence of consecutive characters.

clocking. In data communications, a method of controlling the number of data bits sent on a communications line in a given time.

cluster. Any configuration of interconnected workstations for the purpose of sharing resources (e.g. Local Area Networks, host attached workstations, etc.)

command. A request to perform an operation or execute a program. When parameters, arguments, flags, or other operands are associated with a command, the resulting character string is a single command.

command name. (1) The first or principal term in a command. A command name does not include parameters, arguments, flags, or other operands. (2) The full name of a command when an abbreviated form is recognized by the computer (for example, print working directory for pwd).

communications adapter. A hardware feature enabling a computer or device to become a part of a data communications network.

configuration. The group of machines, devices, and programs that make up a data processing system. See also *system customization.*

coupler. A device connecting a modem to a telephone network.

customize. To describe (to the system) the devices, programs, and users for a particular data processing system.

cylinder. All disk or diskette tracks that can be read or written without moving the disk drive or diskette drive read/write mechanism.

cursor. (1) A movable symbol (such as an underline) on a display, used to indicate to the operator where the next typed character will be placed or where the next action will be directed. (2) A marker that indicates the current data access location within a file.

daemon. A process begun by the super user or its shell that can be stopped only by the super user. Daemon processes generally provide services that must be available at all times such as sending data to a printer.

data communications. The transmission of data between computers, or remote devices or both (usually over long distance).

data link. The equipment and rules (protocols) used for sending and receiving data.

data stream. All information (data and control information) transmitted over a data link.

debug. (1) To detect, locate, and correct mistakes in a program. (2) To find the cause of problems detected in software.

debugger. A device used to detect, trace, and eliminate mistakes in computer programs or software.

default. A value that is used when no alternative is specified by the operator.

default value. A value stored in the system that is used when no other value is specified.

device driver. A program that operates a specific device, such as a printer, disk drive, or display.

device manager. Collection of routines that act as an intermediary between device drivers and virtual machines for complex interfaces. For example, supervisor calls from a virtual machine are examined by a device manager and are routed to the appropriate subordinate device drivers.

dialog. The interchange of information between two people or between a person and a computer by questions and answers.

duplex. Pertains to communications data that can be sent and received at the same time. Same as *full duplex.* Contrast with *half duplex.*

editor. A program used to enter and modify programs, text, and other types of documents.

escape character. The character that changes the meaning of the characters that follow. For example, the backslash character is used to indicate to the shell that the next character is not intended to have the special meaning normally assigned to it by the shell.

file. A set of related records treated as a unit.

flag. A modifier that appears on a command line with the command name that modifies the action of the command. Flags in AIX almost always are preceded by a dash. Most commands permit the user to omit all flags.

foreground. A mode of program execution in which the shell waits for the program specified on the command line to complete before returning your prompt.

format. (1) A defined arrangement of such things as characters, fields, and lines, usually used for displays, printouts, or files. (2) To arrange such things as characters, fields, and lines.

group ID number. A unique number assigned to a group of related users. The group number can often be substituted in commands that take a group name as an argument.

half duplex. Pertains to communications in which data can be sent in only one direction at a time. Contrast with *duplex*.

hexadecimal. Pertaining to a system of numbers to the base sixteen; hexadecimal digits range from 0 (zero) through 9 (nine) and A (ten) through F (fifteen).

high-order. Most significant; leftmost. For example, bit 0 in a register.

home directory. (1) A directory associated with an individual user. (2) The user's current directory on login or after issuing the cd command with no argument.

input file. A file opened in the input mode.

input-output file. A file opened in the I-O mode.

interactive processing. A processing method in which each operator action causes response from the program or the system. Contrast with *batch processing*.

interface (n). A shared boundary between two or more entities. An interface might be a hardware component to link two devices together or it might be a portion of storage or registers accessed by two or more computer programs.

interrupt. (1) To temporarily stop a process. (2) In data communications, to take an action at a receiving station that causes the sending station to end a transmission. (3) A signal sent by an I/O device to the processor when an error has occurred or when assistance is needed to complete I/O. An interrupt usually

suspends execution of the currently executing program.

kill character. The character that is used to delete a line of characters entered after the user's prompt.

licensed programs. Software programs that remain the property of the manufacturer, for which customers pay a license fee.

local. Pertaining to a device directly connected to your system without the use of a communications line. Contrast with *remote.*

log. To record; for example, to log all messages on the system printer. A list of this type is called a log, such as an error log.

log in. To begin a session at a display station.

low-order. Least significant; rightmost.

mailbox. An area designated for storage of mail messages directed to a specific system user.

mask. A pattern of characters that controls the keeping, deleting, or testing of portions of another pattern of characters.

modulation. Changing the frequency or size of one signal by using the frequency or size of another signal.

modulator-demodulator (modem). A device that converts data from the computer to a signal that can be transmitted on a communications line, and converts the signal received to data for the computer.

multi-user environment. A computer system that provides terminals and keyboards for more than one user at the time.

network. A collection of products connected by communication lines for information exchange between locations.

node. An individual element of a full pathname. Nodes are separated by slashes (/).

nonswitched line. A connection between computers or devices that does not have to be established by dialing. Contrast with *switched line.*

online. Being controlled directly by, or directly communicating with, the computer, or both.

overflow condition. A condition that occurs when a portion of the result of an operation exceeds the capacity of the intended unit of storage.

password. A string of characters that, when entered along with a user identification, allows an operator to sign on to the system.

password security. A program product option that helps prevent the unauthorized use of a display station, by checking the password entered by each operator at sign-on.

pathname. A complete filename specifying all directories leading to that file.

permission code. A three-digit octal code, or a nine-letter alphabetic code, indicating access permissionsto a file or directory. Access permissions are read, write, and execute.

permission field. One of the three-character fields within the permissions column of a directory listing indicating the read, write, and run permissions for the file or directory owner, group, and all others.

polling. A method for determining whether each of the stations sharing a communications line has data to send.

port. (1) A part of the system unit or remote controller to which cables for display stations and printers are attached. (2) To transfer programs from one computer to another.

process. (1) A sequence of discrete actions required to produce a desired result. (2) An entity receiving a portion of the processor's time for executing a program. (3) An activity within the system begun by entering a command, running a shell program, or being started by another process.

process ID. A unique number assigned to a process that is running.

profile. Data describing the significant features of a user, program, or device.

protocol. In data communications, the rules for transferring data.

protocol procedure. A process that implements a function for a device manager. For example, a virtual terminal manager may use a protocol procedure to interpret the meaning of keystrokes.

public data network. A communications common carrier network providing data communications services over switched or nonswitched lines. an immediate message display.

recovery procedure. (1) An action performed by the operator when an error message appears on the display screen. Usually this action permits the program to continue or permits the operator to run the next job. (2) The method of returning the system to the point where a major system error occurred and running the recent critical jobs again.

run-time environment. A collection of subroutines that provide commonly used functions for system components.

sector. (1) An area on a disk track or a diskette track reserved to record information. (2) The smallest amount of information that can be written to or read from a disk or diskette during a single read or write operation.

server. A program that handles protocol, queuing, routing, and other tasks necessary for data transfer between devices in a computer system.

session. The period of time during which programs or devices can communicate with each other.

shell program. The command interpreter providing the user with an interface to the AIX kernel.

shell procedure. A series of commands combined in a file that carry out a particular function when the file is run or when the file is specified as an argument to the *sh* command. Shell procedures are frequently called shell scripts.

special character. A character other than an alphabetic or numeric character. For example, *, +, >, and % are special characters.

spool file. A disk file containing output that has been saved for later printing.

stand-alone work station. A work station that can be used to preform tasks independent of (without being connected to) other resources such as servers or host systems.

super user (SU). The user who can operate without the restrictions designed to prevent data loss or damage to the system. (User ID 0).

switched line. In data communications, a connection between computers or devices established by dialing. Contrast with *nonswitched line.*

synchronous. Occurring in a regular or predictable sequence.

synchronous data link control (SDLC). A form of communications line control using commands to control the transfer of data over a communications line. Compare with *binary synchronous communications.*

synchronous transmission. In data communications, a method of transmission in which the sending and receiving of characters is controlled by timing signals. Contrast with *asynchronous transmission.*

system user. A person, process, or other resource that uses the facilities of a computer system.

systems network architecture (SNA). A set of rules for controlling the transfer of information in a data communications network.

trace. To record data that provides a history of events occurring in the system.

track. A circular path on the surface of a disk or diskette on which information is magnetically recorded and from which recorded information is read. billing are examples of transactions.

transfer. To move data from one location to another in a computer system or between two or more systems.

transmission control characters. In data communications, special characters that are included in a message to control communication over a data link. For example, the sending station and the receiving station use transmission control characters to exchange information; the

receiving station uses transmission control characters to indicate errors in data it receives.

user identification (user ID). A unique string of characters identifying an operator to the system. This string of characters limits the functions and information the operator is allowed to use. The user's ID can often be substituted in commands that take a user's login name as an argument.

user list. A list, containing the user identification and access levels, of all operators who are allowed to use a specified file or library.

user profile. A file containing a description of user characteristics and defaults (for example, printer assignment, formats, group ID) to be conveyed to the system while the user is signed on.

voice-grade telephone line. A telephone line normally used for transmission of voice communications. The line requires a modem for data communications.

Index

Special Characters

! in path name (uucp) 7-9
! subcommand (tcp/ip) 4-10
/dev directory
  change settings B-4
  check listings B-4
  create port B-4
/etc/portstatus file 9-5
/usr/INnet/connect.con file 3-9
/usr/INnet/connect.con.t file 3-9
/usr/lib/dir 5-26
/usr/lib/INnet/dialers file 9-17
/usr/lib/uucp file B-13
/usr/lib/uucp/FWDFILE 8-6, 8-17
/usr/lib/uucp/L-devices file 8-6, 8-10
/usr/lib/uucp/L-dialcodes file 8-6, 8-11
/usr/lib/uucp/L.sys file 7-5, 8-6, 8-12
  fields 8-13
  sample entries 8-16
/usr/lib/uucp/ORIGFILE 8-6, 8-17
/usr/lib/uucp/QTCMDS file 7-20, 8-7, 8-20
/usr/lib/uucp/SQFILE 8-21
/usr/lib/uucp/USERFILE 8-7, 8-18
/usr/mail/name file 2-21
/usr/name/dead.letter file 2-21
/usr/name/mbox file 2-21
/usr/spool/uucp/ERRLOG B-16
/usr/spool/uucp/L_sub file 7-26
/usr/spool/uucp/LOGFILE 7-23
/usr/spool/uucppublic file 7-9, 7-14
~in path name (uucp) 7-9

A

a subcommand (tn) 4-20
adapters 9-7
  problems B-3
alter (ate)
  command 6-9
  menu 6-10
append subcommand (tcp/ip) 4-10
ASCII BEL character 2-6
ascii subcommand (tcp/ip) 4-10
asks for status (tn) 4-20
ate
  customizing 6-3
  installing 5-4
  task overview 5-4
ate command 5-14
ate session
  breaking 5-32
  current settings 5-13
  disconnecting 5-32
  ending 5-32
  leaving 5-36
  making connection 5-14
  prerequisite tasks 5-12
  starting 5-14
  starting program 5-14
ate.def file 6-17, 6-18
attempts
  ate file transfer 6-13
auto answer setting B-6
automatic dialing (ate) 5-17

B

background information ix
background process 7-4
base system program
  installing commands 2-3
binary subcommand (tcp/ip) 4-10
bps setting B-5
break command (ate) 5-32
buffer overflow messages B-6

C

c subcommand (tn) 4-20
cable
  problems B-3, B-5
cables 9-8
call-in
  line 7-4
  port 9-13
    adding a device 9-14
    connection characteristics (example) 9-13
    customizing 9-13
    settings 9-15
  site 1-6
call-out B-5
  line 7-4
  port 9-9, B-5
    adding a device 9-11
    connection characteristics (example) 9-9
    customizing 9-9
    settings 9-12
  site 1-6
capture key (ate) 5-9, 6-15
carriage return/linefeed combinations 6-24
cat command 3-18, 5-35
  from ate 5-35
cd subcommand (tcp/ip) 4-10
changing
  control keys (ate) 6-15
  default file (ate) 6-17
  directory permissions (ate) 5-24
character pacing (ate) 6-14, 6-23
chmod command
  changing directory permissions (ate) 5-24
chparm command 8-5
close
  telnet connection 4-20
command
  related problems B-10
command files (uucp) 7-32
  entries
    receiving files (uucp) 7-35
    sending files (uucp) 7-34
commands (ate)
  alter 6-9
    device 6-13
    final 6-13
    initial 6-13
    length 6-12
    parity 6-12
    rate 6-12
    stop 6-12
    wait 6-13
  break 5-32
  change connection settings 6-9
  change data transmission characteristics 6-9
  change local settings 6-4
  connect 5-14
  directory 5-21
  help 5-33
  modify 6-4
    echo 6-7
    linefeeds 6-7
    name 6-7
    VT100 6-8

write 6-8
Xon/Xoff 6-8
perform 5-35
quit 5-36
receive 5-30
running shell command 5-35
send 5-28
terminate 5-32
using 5-8
commands (port) 9-4, 9-5, 9-6
devices 9-4, 9-6
pdisable 9-5
penable 9-5
phold 9-5
commands (uucp)
uuclean 7-24
uucp 7-7
uudemon 7-28
uulog 7-23
uuname 7-5
uupick 7-14, 7-17
uusub command 7-26
uuto 7-14, 7-15
uux 7-20
common carrier line 9-4
compatible systems
with uucp 7-5
conference
accept a call 2-11
arranging 2-11
being excused 2-13, 2-15
closing 2-17
contending for the floor 2-13
holding 2-12
issue a call 2-11
joinconf 2-12
name 2-12
off the record 2-11, 2-12
on the record 2-11, 2-12
participating 2-13
taking the floor 2-13
transcript 2-16
withdrawing (BYE) 2-14
yielding the floor 2-13
connect
command 3-13
copying files 3-18
ending a session 3-22
escape to local prompt 3-15
running local commands 3-17
stanza 3-8
starting a session 3-13
subcommand
i (include) 3-18
t (transcript) 3-20
to remote system 3-12
using 3-12
connect (ate) 5-14
automatic dialing 5-17
direct connection 5-19
manual dialing 5-15
types 5-14
connect.con file 3-9
connect.con.t file 3-9
connected main menu (ate) 5-7
ctrl-r 5-11
connection
failure B-3
related problems B-10
retaining 2-8
statistics (uucp) 7-26
connection (ate)
settings
altering 6-11
problems B-3
control characters
pacing protocol B-8
control keys (ate)
capture key 5-9, 6-15
changing 6-15
ctrl-b 5-9
ctrl-r 5-10
ctrl-v 5-9
failure B-7

functions 5-9
main-menu key 5-9, 6-15
previous-screen key 5-10, 6-15
remapping 6-15
controlling file access (uucp)
uupick command 7-14
uuto command 7-14
conventions
enter viii
EOF viii
printing viii
conversation
symbol at end 2-7
copy command 5-24
copying files
connect 3-18
cp command 5-24
cp/m files 6-24, B-8
creating
dialing directory (ate) 5-23
cron command 7-26, 7-28
crontab file 7-26, 7-28
ctrl-b (ate) 5-9, 6-15
ctrl-d 2-7
ctrl-r (ate) 5-10, 6-15
ctrl-v (ate) 5-9, 6-15
ctrl-x 6-20, B-9

D

data files (uucp) 7-32
data transmission (ate)
attempts 6-13
characteristics
altering 6-11
pacing protocol 6-13, 6-14
transfer protocol 6-13, 6-14
xmodem protocol 6-13
default file (ate) 6-17
changing values 6-18
editing 6-18
initial values 6-18
defining outgoing physical links (uucp) 8-10
defining the consoles (tcp/ip)
emulate variable 4-17
environment variable 4-17
delete subcommand (tcp/ip) 4-10
deleting mail 2-24
device command (ate) 6-13
device configuration error B-17
device driver problems B-20
device settings
bps B-5
dvam B-5
ixp B-5
nosb B-5
pt B-5
rts B-5
sns B-5
tt B-5
devices command 9-4, 9-6
dialing (ate)
automatic 5-17
failure B-3
manual 5-15
dialing directory (ate)
creating 5-23
displaying 5-21
format 5-26
dialing directory file (ate)
field descriptions 5-26, 5-27
diff command 7-22
dir subcommand (tcp/ip) 4-10
direct connection (ate) 5-19
directory (ate)
changing permissions 5-24
command 5-21
fields
echo 5-27
length 5-27
linefeed 5-27

name 5-26
number 5-26
parity 5-27
rate 5-26
stop 5-27
file 5-23, 5-26, 5-27
menu 5-26
directory (tcp/ip)
contents 4-11
disconnecting
ate session 5-32
displaying
dialing directory (ate) 5-21
dos files 6-24, B-8
dvam setting B-5

E

echo command (ate) 5-27, 6-7
emulate variable (tcp/ip) 4-17
end of conversation 2-7
end of file symbol (EOF) 2-7
end of message 2-7
ending
ate session 5-32
ending a local message 2-6
enter viii
environment variable (tcp/ip) 4-17
emulate 4-17
EOF viii
equipment problems B-17
ERRLOG B-16
establishing subnetworks (uucp) 7-26
Ethernet
communication on 4-3
execute file (uux)
command line 7-39
mail suppression line 7-39
required file line 7-38
script 7-38
standard input line 7-38
standard output line 7-38
user line 7-38
execute files (uucp) 7-32
exiting ate 5-36

F

file access (uucp) 8-18
file transfer (ate)
default file 6-17
failure using pacing protocol B-8
pacing protocol 6-23
protocol
character pacing 6-14
integer pacing 6-14
pacing 6-13
xmodem 6-13
xmodem protocol 6-20
file transfer (tcp/ip)
! subcommand 4-10
append subcommand 4-10
ascii subcommand 4-10
binary subcommand 4-10
cd subcommand 4-10
delete subcommand 4-10
dir subcommand 4-10
get subcommand 4-10
help subcommand 4-11
lcd subcommand 4-11
ls subcommand 4-11
mget subcommand 4-11
mput subcommand 4-11
put subcommand 4-11
pwd subcommand 4-11
quit subcommand 4-11
rename subcommand 4-11
xftp command 4-8
file transfer (uucp) 7-7
background process 1-6

failure B-12
local 7-10
remote 7-12
final command (ate) 6-13
finger command 4-5
how to use 4-6
forwarding
files (uucp) 7-12
mail 2-20

G

get subcommand (tcp/ip) 4-10
getty -d B-17
giving commands (ate) 5-8

H

handling the mail 2-22
hardwiring 9-4
help command (ate) 5-33
help subcommand (tcp/ip) 4-11
high-speed line problems B-21

I

i (include) subcommand 3-18
information on users 4-5
initial command (ate) 6-13
installing
aix A-4
communication facilities A-5
minidisk space A-3
ate 5-4, A-5
commands A-5
base system program 2-3
inter-workstation 2-3, A-5
multi-user services 2-3, A-5
tcp/ip 4-3, A-5
uucp 7-4, A-5
integer pacing (ate) 6-14
inter-workstation
installing commands 2-3
inter-workstation commands
creating local communication facility 2-4
interval pacing 6-23
issuing commands (ate) 5-8
ixp setting B-5

J

joinconf 2-12

K

kapture file (ate) 6-7

L

L.sys file 7-5
sample entries 8-16
L.sys file fields 8-13
lcd subcommand (tcp/ip) 4-11
leaving ate 5-36
length command (ate) 5-27, 6-12
li command B-4
line
call-in 7-4
call-out 7-4
line noise B-17
line protocol B-14
linefeed/carriage return characters

using xmodem protocol B-8
linefeeds command (ate) 5-27, 6-7
local communication facility
creating 2-4
messages
sending 2-6
who can receive 2-5
who command 2-5
write command 2-6
types 2-4
with base system program
mail command 2-4
write command 2-4
with multi-user services
confer command 2-4
mesg command 2-4
who command 2-4
lock files B-13
cleaning B-16
log entry file (uucp) 8-22
LOG file 8-22
LOGFILE (uucp) 7-23
login
remote (tcp/ip)
sequence (uucp) 8-14
login sequence (uucp)
escape sequences 8-15
losing characters on screen B-6
ls subcommand (tcp/ip) 4-11

M

mail
ending the message 2-19
forwarding 2-20
handling 2-22
intermediate systems 3-5
reading 2-20
receiving 2-20
remote 3-4
remote mail 2-19
sending 3-4
mail command 3-4
tcp/ip 4-14
sending to several users 2-19
mail files
/usr/mail/name 2-21
/usr/name/dead.letter 2-21
/usr/name/mbox 2-21
main-menu key (ate) 5-9, 6-15
making a connection (ate) 5-14
manual dialing (ate) 5-15
menus (ate)
alter 6-10
connected 5-7
directory 5-26
modify 6-4, 6-5
unconnected 5-5
mesg
changing default 2-10
editing profile 2-10
mesg n 2-9
mesg y 2-9
superuser override 2-9
messages
end of file symbol (EOF) 2-7
ending 2-7
in files 2-7
longer 2-7
receiving 2-9, 2-10
start-up procedure 2-10
rejecting 2-9
sending 2-6
status 2-9
who can receive 2-5
who command 2-5
write command 2-6
mget subcommand (tcp/ip) 4-11
modem 9-4
autodialer programs 9-16, 9-17, 9-18
invoking 9-17
phone numbers 9-18

return codes 9-17
sample program 9-18
hang-up failure B-7
problems B-3, B-5
settings 9-16
modem eliminator 9-4, 9-8
modems
customizing
call-in port 9-13
call-out port 9-9
external 9-9
internal 9-9
modify (ate)
command 6-4
menu 6-5
subcommands 6-7
mput subcommand (tcp/ip) 4-11
multi-user services
creating local communication facility 2-4
installing commands 2-3

N

n subcommand (tn) 4-20
name command (ate) 5-26, 6-7
netmail command 4-14
network
communication on 4-3
customization 4-4
installing 4-3
management 4-4
next node (uucp) 8-17
no answer B-5
nonsense on screen B-6, B-9
nosb setting B-5
null modem 9-4, 9-8
number command (ate) 5-26

O

o (over) 2-13
oo (over and out) 2-13
options with uupick 7-20
ordering manuals x

P

pacing protocol 6-13, 6-14
character 6-14
file transfer (ate) 6-23
integer 6-14
parity command (ate) 5-27, 6-12
path names
uucp 7-9
uux 7-22
pdisable command 9-5
penable command 9-5
perform command (ate) 5-35
permissions 7-11
phold command 9-5
phone line
problems B-4, B-5
phone number prefixes (uucp) 8-11
ping command 4-7
port
file doesn't open B-17
port commands 9-5
port failures
diagnosing B-17
tracing B-18
ports 9-4
call-in 9-5
call-out 9-4
commands 9-4
configuring 9-6
connecting
adapters 9-7

cables 9-8
modem eliminator 9-8
null modem 9-8
connections 9-4
customizing 9-4
failure to open B-4
setting up 9-6
postmark 2-21
prerequisite information ix
previous process running B-17
previous-screen key (ate) 5-10, 6-15
primary site 9-4
printing conventions viii
problem determination
asynchronous terminal emulation
buffer overflow messages B-6
control keys failure B-7
dialing failure B-3
during a connection B-6
establishing a connection B-3
file transfer failure B-8
incorrect linefeed/carriage return characters B-8
losing characters B-6
modem doesn't hang up B-7
no answer B-5
nonsense on screen B-6, B-9
opening a port B-4
using pacing protocol B-8
using xmodem protocol B-8
uucp B-10
checking previous commands B-11
checking previous connections B-10
device configuration error B-17
device driver problems B-20
diagnosing port failures B-17
equipment problems B-17
file transfer failure B-12
high-speed line problems B-21
line noise B-17
port file doesn't open B-17
previous process running B-17
tracing port failures B-18
transmission problems B-20
using uucico command B-12
using uustat command B-10
profile
changing default
mesg 2-10
prompt
escape to local 3-15
psubcommand (tn) 4-20
pt setting B-5
PUBDIR (uucp) 7-9
in controlling file access 7-14
public directory (uucp) 7-9
in controlling file access 7-14
put subcommand (tcp/ip) 4-11
pwd subcommand (tcp/ip) 4-11

Q

QTCMDS file 7-20
quit
telnet 4-20
tn 4-20
quit command (ate) 5-36
quit subcommand (tcp/ip) 4-11

R

rate command (ate) 5-26, 6-12
reading
mail 2-20
receive command (ate) 5-30
receive command-files (uux) 7-38
receiving
mail 2-20
receiving files
for a specific ID (uupick) 7-18

uucp 7-7, 7-10, 7-13
command-file fields 7-35
for a specific ID (uupick) 7-17
from a remote system 7-35
remote transfer 7-13
xmodem command 6-22
redirection characters
uux 7-22
remapping control keys (ate) 6-15
remote commands (uucp) 8-20
remote communication facility
creating 3-3
types 3-3
with base system program 3-3
remote file transfer 7-12
remote login
tcp/ip 4-17
defining the consoles 4-17
remote login variable (tcp/ip) 4-17
term 4-17
remote mail 2-19
remote system
connecting to 3-12
copying files to
connect 3-18
remote system names 7-5
removing old files (uucp) 7-24
uux command 7-24
rename subcommand (tcp/ip) 4-11
requesting
remote system status 4-7
user information 4-5
return to shell (tn) 4-20
returning to operating system
from ate 5-36
routing through intermediate systems 3-5
rts setting B-5
running commands automatically
(uucp) 7-28
uudemon command 7-28
running commands remotely
uucp 7-7
uux 7-20
running shell command
from ate 5-35

S

sample autodialer program 9-19
sample directory file 5-26
secondary site 9-5
security (uucp)
file access 8-18
logins 8-16
send command (ate) 5-28
sending
local mail 2-6, 2-18, 2-19
several mail boxes 2-18, 2-19
local messages 2-6
sending files
uucp 7-7, 7-10, 7-12
/usr/lib/uucp/QTCMDS file 7-36
command-file fields 7-34
from a remote system 7-36
local transfers 7-10
remote transfers 7-12
through remote systems 7-12
to a remote system 7-34
to a specific ID (uuto) 7-15
uucico program 7-36
uucp -e 7-36
xmodem command 6-21
sending mail
tcp/ip 4-14
sequence check file (uucp) 8-21
shell commands
cat 5-35
chmod 5-24
cp 5-24
shell redirection characters
uux 7-22
sns setting B-5

source file 7-10
source of forward request (uucp) 8-17
special shell characters 7-10
spool work file fields
  grade B-12
  number B-12
  system grade B-12
  type B-12
starting ate 5-14
status information (uucp)
  uulog command 7-23
stop command (ate) 5-27, 6-12
STST file 8-22
subnetworks (uucp)
  establishing 7-26
  uusub 7-26
suspend tn 4-20
system names (uucp)
  for call-in sites 8-12
  for call-out sites 8-12
  sample entries 8-16
system prompt ($) 5-36
system status file (uucp) 8-22

T

t (transcript) subcommand 3-20
tasks
  UUCP 1-6
tasks (ate)
  overview 5-4
tcp/ip 4-18
  customizing 4-4
    devices command 4-4
    route command 4-4
  finger command 4-5
  how to use 4-19
  information on specific user 4-5
  installing 4-3
  list of current users 4-5
  netmail command 4-14, 4-16
    how to read your mail 4-16
    how to send a file 4-14
    how to send a note 4-14
  overview 4-4
  ping command 4-7
  remote login 4-17
  remote systems status 4-7
  subcommands 4-20
  Telnet 4-4
  tn command 4-17, 4-18, 4-19, 4-20
  with AIX 4-3
  xftp command 4-8, 4-10
    ! subcommand 4-10
    append subcommand 4-10
    ascii subcommand 4-10
    binary subcommand 4-10
    cd subcommand 4-10
    delete subcommand 4-10
    dir subcommand 4-10
    get subcommand 4-10
Telnet 4-4
telnet protocol
  subcommands 4-20
temporary data file (uucp) 8-21
term variable (tcp/ip) 4-17
terminate command (ate) 5-32
terminating a connection
  local communication 2-6
TM file 8-21
tn command (tcp/ip) 4-17, 4-20
  how to use 4-19
  subcommands 4-20
    ? 4-20
    a 4-20
    ask status 4-20
    b 4-20
    c 4-20
    close connection 4-20
    display help message 4-20
    display status 4-20
    e 4-20

n 4-20
p 4-20
q 4-20
quit 4-20
return to shell 4-20
s 4-20
send break command 4-20
transmit character by character 4-20
transmit line by line 4-20
suspend 4-20
traffic
statistics (uucp) 7-26
transcript
remote connection 3-20
viewing remote 3-22
transcript of conference 2-16
transfer protocol 6-13
pacing 6-13
xmodem 6-13
transferring files
tcp/ip 4-8
xftp command 4-8
uucp 7-7
local transfers 7-10
remote transfers 7-12
transferring files (tcp/ip) 4-12
how to 4-12
transmission problems B-20
tt setting B-5

U

unconnected main menu (ate) 5-5
ctrl-r 5-10
using Asynchronous Terminal Emulation
directory menu 5-26
starting the program 5-14
using intermediate systems
uucp 7-12
uucico command (uucp) B-12, B-15
flags B-15
uucico program 7-32
uuclean command (uucp) 7-24
uucp
administrative files
/usr/lib/uucp/SQFILE 8-21
log entry file 8-22
LOG file 8-22
sequence check file 8-21
STST file 8-22
system status file 8-22
temporary data file 8-21
TM file 8-21
communication characteristics
bits per second 8-7
callback 8-7
commands (remote) 8-7
device name 8-7
dialing code 8-7
dialing device 8-7
dialing prefix 8-7
login name (remote) 8-7
login sequence (remote) 8-7
name (remote system) 8-7
path name 8-8
phone number 8-8
system name (forward for) 8-8
system name (forward to) 8-8
system name (remote) 8-8
telephone number 8-8
time 8-8
transmission speed 8-8
type of link 8-8
customizing 8-4
defining communication characteristics 8-6, 8-8
defining outgoing physical links 8-10
editing files 8-6, 8-8
file access 8-18
next node 8-17

phone number prefixes 8-11
remote commands 8-20
source of forward request 8-17
system names 8-12
directories
/usr/lib/uucp 7-33
/usr/spool/uucp 7-33
/usr/spool/uucp/.XQTDIR 7-33
/usr/spool/uucppublic 7-33
file transfer
uucico program 7-32
uucp program 7-32
files
/usr/lib/uucp/FWDFILE 8-6, 8-17
/usr/lib/uucp/L-devices 8-6, 8-10
/usr/lib/uucp/L-dialcodes 8-6, 8-11
/usr/lib/uucp/L.sys 8-6, 8-12
/usr/lib/uucp/ORIGFILE 8-6, 8-17
/usr/lib/uucp/QTCMDS 8-7, 8-20
/usr/lib/uucp/USERFILE 8-7, 8-18
data 7-32
execute 7-32
work (command) 7-32
how process works 7-32, 7-37
installing 7-4
overview 7-4
primary tasks 7-32
running remote commands 7-32
uux program 7-32
uuxqt program 7-32
uucico program 7-37
uuxqt program 7-37
uucp command (uucp) 7-7, 7-10, 7-12
local file transfer 7-10
path names 7-9
remote file transfer 7-12
uucp program 7-32
uudemon command (uucp) 7-28
uudemon.day (uucp) 7-30
uudemon.hr (uucp) 7-29
uudemon.week (uucp) 7-31
uulog command (uucp) 7-23
uuname command 7-5
uupick command (uucp) 7-17, 7-18
options 7-20
uustat command (uucp) B-10
uuto command (uucp) 7-15
uux command (uucp) 7-20
uux program 7-32
uuxqt program 7-32

V

VT100 command (ate) 6-8

W

wait command (ate) 6-13
who command
local communications 2-5
work file entries
See command files (uucp)
work files (uucp) 7-32
write
redirecting write command 2-8
write command 2-6
how to use 2-6
local communications 2-6
write command (ate) 6-8

X

xftp command 4-8
making the connection 4-9
subcommands 4-10, 4-11
! 4-10
append 4-10

appending file on remote system 4-10
ascii 4-10
binary 4-10
cd 4-10
changing file name on remote system 4-11
changing local working directory 4-11
changing remote working directory 4-10
delete 4-10
deleting remote file 4-10
dir 4-10
displaying contents of remote directory 4-10, 4-11
displaying help message 4-11
displaying name of current remote directory 4-11
exiting program 4-11
get 4-10
help 4-11
invoking shell 4-10
lcd 4-11
ls 4-11
mget 4-11
mput 4-11
put 4-11
pwd 4-11
quit 4-11
rename 4-11
setting ASCII transfer type 4-10
setting binary-image transfer 4-10
storing local file on remote system 4-11
storing local files on remote system 4-11
storing remote file on local system 4-10
storing remote files on local system 4-11
xmodem command
  interrupting 6-20
  receiving files 6-20, 6-22
  sending files 6-20, 6-21
xmodem protocol 6-13
  file transfer 6-20
Xon/Xoff command (ate) 6-8

IBM RT PC
Communications Family

Reader's Comment Form

AIX Operating System
Communications Guide

SC23-0791-0

Your comments assist us in improving our products. IBM may use and distribute any of the information you supply in any way it believes appropriate without incurring any obligation whatever. You may, of course, continue to use the information you supply.

For prompt resolution to questions regarding set up, operation, program support, and new program literature, contact the authorized IBM RT PC dealer in your area.

Comments:

NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES

BUSINESS REPLY MAIL

FIRST CLASS PERMIT NO. 40 ARMONK, NEW YORK

POSTAGE WILL BE PAID BY ADDRESSEE

International Business Machines Corporation
Department 997, Building 998
11400 Burnet Rd.
Austin, Texas 78758

Fold and tape

Fold and tape

Cut or Fold Along Line

Tape

Please Do Not Staple

Tape

IBM RT PC AIX Operating System Communications Guide — **SC23-0791**

Book Title — **Order No.**

Book Evaluation Form

Your comments can help us produce better books. You may use this form to communicate your comments about this book, its organization, or subject matter, with the understanding that IBM may use or distribute whatever information you supply in any way it believes appropriate without incurring any obligation to you. Please take a few minutes to evaluate this book as soon as you become familiar with it. Circle Y (Yes) or N (No) for each question that applies and give us any information that may improve this book.

Y N Is the purpose of this book clear?

Y N Is the table of contents helpful?

Y N Is the index complete?

Y N Are the chapter titles and other headings meaningful?

Y N Is the information organized appropriately?

Y N Is the information accurate?

Y N Is the information complete?

Y N Is only necessary information included?

Y N Does the book refer you to the appropriate places for more information?

Y N Are terms defined clearly?

Y N Are terms used consistently?

Y N Are the abbreviations and acronyms understandable?

Y N Are the examples clear?

Y N Are examples provided where they are needed?

Y N Are the illustrations clear?

Y N Is the format of the book (shape, size, color) effective?

Other Comments

What could we do to make this book or the entire set of books for this system easier to use?

Optional Information

Your name

Company name

Street address

City, State, ZIP

No postage necessary if mailed in the U.S.A.

Tape

Please Do Not Staple

Tape

Cut or Fold Along Line

Fold and tape

Fold and tape

NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES

BUSINESS REPLY MAIL

FIRST CLASS PERMIT NO. 40 ARMONK, NEW YORK

POSTAGE WILL BE PAID BY ADDRESSEE

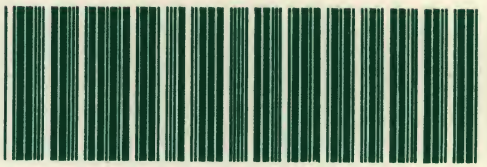
International Business Machines Corporation
Department 997, Building 998
11400 Burnet Rd.
Austin, Texas 78758

International Business
Machines Corporation
Department 997, Building 998
11400 Burnet Rd.
Austin, Texas 78758

Printed in the
United States of America

SC23-0791-0

SC23-0791-00



92X1263